THE MAGAZINE FOR AMIGA PROGRAMMERS
FEBRUARY 1989: VOLUME 1, ISSUE 5

Canada $4.25
USA $3.50

Amiga Transactor FOR THE

- Professional animation tips *by Derek Grime*
- Algorithmic solid model generation *by Cathryn and Eric Graham*
- Devices and Libraries *by Betty Clay*
- ROBBS: Rexx Object Building Block System *by Larry Phillips*
- Modula-2 programming, part 2 *by Clark Williams*
- Using TDI's M2AMIGA *by Anthony Bryant*
- Online lingo: communication subtleties *by James D. Williams*
- Book Reviews: *The C Programming Language*, 2nd Edition; *Tips and Tricks*
- Product Reviews: Ultra DOS utilities, AmigaTEX, Comic Setter, Pixmate
- **Plus** regular columns by Larry Phillips, Don Curtis and Steve Ahlstrom; new products and more.

The Chalice of Janus by Louis Markoya

0 29074 12122 02

WIN A MONITERM!

Build an expert system application with the MAGELLAN™ artificial intelligence expert system building tool and win a MONITERM Viking I high resolution monitor for your AMIGA.
Imagine the possibilities: the power of the AMIGA coupled with the flexibility of MAGELLAN!

Categories for the First Annual MAGELLAN Application Contest are:

Most Productive Application
Business or Diagnostic systems, decision assistance tools for financial investment, production control systems, or inventory management systems.

Most Creative Application.
Music or Animation systems, or innovative approaches to traditional expert systems.

Most Bizarre Application.
Let your imagination run wild!

3 Grand Prize Winners will receive:
A Moniterm Monitor
A MAGELLAN Developer Package
Retail Value $3000.

2 Runners-Up in each category will receive:
MAGELLAN Developer Package
Retail Value $1000.

All finalists will also receive a MAGELLAN designer sweatshirt.

All entries must be received by Emerald Intelligence by June 1, 1989. Contest void where prohibited. Contest Winners will be announced at Ami-Expo Chicago in July 1989. All entries become the exclusive property of Emerald Intelligence Inc., and cannot be returned. Winning entries may be used in a promotional campaign and any entry may be included in promotional materials or application manuals. For a complete set of rules and an entry form, send a self addressed stamped envelope to MAGELLAN Contest Rules, c/o Emerald Intelligence.

MAGELLAN™

THE EXPERT SYSTEM SOFTWARE

Emerald Intelligence

334 South State Street, Ann Arbor, Michigan 48104 (313) 663-8757

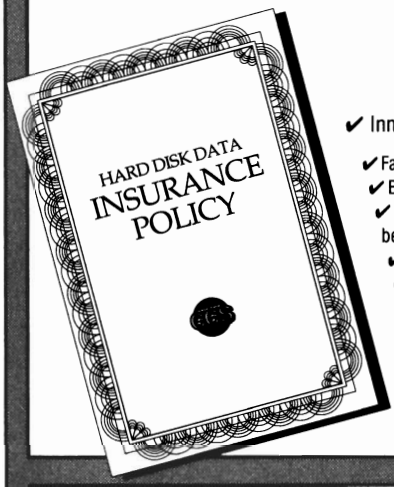
MAGELLAN is a trademark of Emerald Intelligence Inc. AMIGA is a registered trademark of Commodore-Amiga Inc. Moniterm is a trademark of Moniterm Corporation.

FOR THE AMIGA

Protecting your valuable hard disk files is easier and faster than ever before!

QUARTERBACK™

*The **FASTEST** Hard Disk Backup Utility!*



Backup to or restore from:

✔ Floppy Disks ✔ Streaming tape (AmigaDOS-compatible)
✔ CLtd's Konica 10.7MB high-density floppy drive
✔ Inner-Connection's Bernoulli drive ✔ ANY AmigaDOS-compatible devise

✔ Fast backup - 20MB in 30 minutes or less ✔ Uses two floppy drives (if available) for backup/restore with automatic switching ✔ Builds, sorts and displays catalog of files and subdirectories ✔ Provides FULL/Subdirectory/Individual file backup/restore ✔ Includes or excludes files by name (with wild cards), file date, or archive bit ✔ Calculates the number of floppies you'll need before you start ✔ Handles files of unlimited length, unlimited subdirectories and unlimited files per subdirectory ✔ Automatically formats diskettes with no delay as it writes ✔ Sequentially numbers and date/time stamps backup diskettes ✔ Checks the sequence number and date/time stamp of each diskette before restoring files from it ✔ Restores original date/time stamp, file notes, and protection bits on both files and subdirectories ✔ Runs with Workbench or CLI ✔ Produces backup/restore report to disk or printer ✔ Beeps for floppy change ✔ Accepts CLI parameters and batch command files ✔ Detects bad disks during backup or restore ✔ Convenient/user friendly error recovery ✔ Multitasking ✔ Runs in 512K ✔ No copy protection ✔ Works with all AmigaDOS compatible hard disk drives.

Only $69.95 Plus $3.00 shipping and handling, CA residents ass 6% sales tax.

DISK 2 DISK™

Convert C64/C128 Files to the Amiga!

DISK-2-DISK™ makes it easy and convenient to transfer C64/C128 files to and from the Amiga! DISK-2-DISK programs the Amiga model 1020 external 5.25" disk drive to read and write 1541/4040 and 1570/1571 disk formats including 1541 "flippies".

✔ Converts Commodore/PET ASCII to AmigaDOS standard ASCII and vice versa ✔ Transfers word processing text files (such as PaperClip, SpeedScript and Pocket Writer) to and from the Amiga for use with popular Amiga word processors ✔ Includes 3 public domain programs for converting C64 Koala, PrintShop and Doodle files to IFF format ✔ Finds and flags dialect differences between Commodore Basic and Amiga Basic files ✔ Provides VALIDATE BAM and CHECK DISK utilities (VALIDATE BAM verifies the directory structure of the 1541/1571 diskette; CHECK DISK reads every block of a 1541/1571 diskette to detect diskette errors).

DISK-2-DISK requires the Amiga model 1020 5.25" disk drive.

Only $49.95
Plus $3.00 shipping and handling
CA residents add 6% sales tax.

DOS 2 DOS™

DOS-2-DOS transfers MS-DOS and Atari ST files to and from AmigaDOS!

DOS-2-DOS version 3.0 permits access to any MS-DOS volume available via AmigaDOS, including MS-DOS partitions on hard disks and MS-DOS volumes on LANS or SCSI networks.

✔ Supports single and double sided 5.25-inch as well as 3.5-inch 720KB MS-DOS diskettes ✔ Reads/Writes 3.5-inch Atari ST diskettes (GEM format) ✔ Reads a variety of 5.25-inch MS-DOS floppy formats via the CLTD Konica high-density floppy drive ✔ Converts ASCII file line-ending characters and provides Wordstar compatibility ✔ Supports full directory path names, with wild cards in the file names ✔ Allows selection of MS-DOS and AmigaDOS subdirectory and displays sorted directory listing ✔ Formats 3.5-inch and 5.25-inch MS-DOS diskettes and Atari ST diskettes ✔ Provides duplicate file name detection with query/replace options ✔ Provides TYPE and DELETE commands ✔ Permits renaming of files where file name restrictions occur ✔ Remains resident to permit AmigaDOS disk swapping.

Only $55.00
Plus $3.00 shipping and handling
CA residents add 6% sales tax

CCS

Central Coast Software™

424 Vista Avenue, Golden, CO 80401 • Telephone: 303 / 526-1030 • FAX 303 / 526-0520

DEALER INQUIRES WELCOME

VISA

Volume 1, Issue 5

Publisher
Antony Jacobson

Vice-President Operations
Jeannie Lawrence

Vice-President Advertising
Maximillian Jacobson

Assistant Advertising Manager
Julie Cale

Editors
Nick Sullivan
Chris Zamara

Assistant Editor
Malcolm O'Brien

Contributing Writers
Steve Ahlstrom
Charles B. Blish
Anthony Bryant
Jim Butterfield
Betty Clay
Don Curtis
Matt Dillon
John Faichney
Andy Finkel
Brad Fowles
Eric Giguère
Cathryn Graham
Derek Grime
Chris Gray
Louis Markoya
Eugene P. Mortimore
Bryce Nesbitt
Rob Peck
Larry Phillips
Dan Schein
Steve Simpson
John Toebes
Victor A. Wagner
Clark Williams
James D. Williams
David Wood

Cover Artist
Louis Markoya

for the AMIGA Transactor
The Magazine for Amiga Programmers

Solid Modelling 20
by Cathryn and Eric Graham
How to create Sculpt-3D format 'scene' files with a C program; you can create complex objects algorithmically that would be impossible by hand!

Animation Confidential 28
by Derek Grime
The graphics on most Amiga programs is terrific, but animation is typically lifeless and unrealistic. Derek gives some professional animation tips to add life to your creations

Online Lingo 32
by James D. Williams
If you're new to online services, you may wonder what funny language the people online are speaking. James explains some conventions, protocols and textual faces

Introducing ROBBS 34
by Larry Phillips
ROBBS stands for Rexx Object Building Block System. This module is a serial communications utility that can be driven with an ARexx script

A Device Is A Library Is A ...
by Betty Clay
In the Amiga's operating system, libraries and devices are surprisingly similar; here's an introduction to what they are at the OS level

Amiga Modula 2 Programming 40
by Clark Williams
More Modula 2 programming tips on the Amiga

Programming in M2Amiga 48
by Anthony Bryant
A description of the M2Amiga Modula 2 development system

I'd rather switch than fight 51
by Monica Dobrov
One graphic designer's experiences in moving from traditional techniques to the Amiga

Library Name and Management Macros 52
by Eugene P. Mortimore
There are some useful assembler macros defined for you in the standard include files - here is an explanation of some that relate to shared library usage

Departments and Columns

Amiga Dispatches 4

by Don Curtis

Hard drive fixes and tips, and some sage advice from Bryce Nesbitt, who is now working at Commodore-Amiga

ViewPort 12

by Larry Phillips

What's in store for the Workbench in 1.4? Larry looks at some popular wishes

ACCESS - The best of non-commercial software 14

by Steve Ahlstrom

Disk Storage 5.0, HandyIcons, Slideshow construction set, and some notes on CAD programs

NEWS Link 73

Reviews

Comic Setter 56

It's not a game, but boy, is it fun

Ultra DOS Utilities, Module 1 60

Easy to use file utilities integrated in one application. Too bad about the manual...

PIXMATE 62

What is image processing? And why is it making Russ Douglas so giddy?

AmigaTEX 66

Donald Knuth's amazing TEX typesetting system comes to the Amiga - and it looks good!

The C Programming Language, 2nd Edition 70

Kernighan and Ritchie's classic definition of the C programming language is brought up to date

Amiga Tips and Tricks 72

Jim Butterfield thinks this Abacus book is the best - and the worst - Amiga book he's ever read

About the cover: *The Chalice of Janus* by Louis Markoya

Louis Markoya also did the popular "Watching Angel" picture for Volume 1, Issue 2. Louis Markoya is a Computer Graphics artist specializing in Amiga imagery. He can be contacted through Amiga Transactor, on PLINK as L.MARKOYA or on Compuserve as 75766,504. Here is his explanation of the picture:

"This Ray Traced picture, rendered with Turbo Silver, depicts the old standard double image, 'Is it a chalice, or is it two faces?' The Chalice of Janus is named after the Roman mythological god Janus, a solar god with two faces. Chaos took the form of Janus, and Chaos takes form in Computer Graphics.

"The main body of the picture was traced in Turbo Silver, the landscape was created with Terrain. The Chalice is an object created by spinning the traced profile of my daughter Amber Rose, as taken from her shadow. The clouds, bird and tree were added with Photon Paint II and Diamond to complete the features of the 'invisible' faces."

Another note of interest on this cover is our new logo; it looks the same, but was created on the Amiga by Monica Dobrov at Beyond Graphics (Monica also has an article in this issue). This means our entire cover is now in electronic form, on the Amiga, and we can generate complete colour separations using Professional Page. This makes the magazine production 100% Amiga-based, except for final film output, which is done on a Linotron 300 PostScript phototypesetter.

TransAmi's
phone number is:
(416) 764-5273

FAX: (416) 764-9262

TOLL-FREE ORDER LINE
1-800-248-2719 Extension 911
(for orders only; have your VISA or Mastercard number ready; available in the U.S. only)

Transactor for the Amiga is published monthly by Croftward Publishing Inc., 10-85 West Wilmot Street, Richmond Hill, Ontario, L4B 1K7. ISSN# 0838-0163. Canadian Second Class Mail Registration No. 7690, Gateway-Mississauga, Ont. US Second Class mail permit pending at Buffalo, NY. USPS Postmasters: send address changes to: Transactor, PO Box 338, Station C, Buffalo, NY, 14209.

Croftward Publishing Inc. is in no way connected with Commodore Business Machines Ltd. or Commodore Incorporated. Commodore and Commodore product names are registered trademarks of Commodore Inc.

Subscriptions: see order card at centre of magazine for complete ordering information and prices.

Send all subscriptions to: Transactor Subscriptions Dept. 10-85 West Wilmot Street, Richmond Hill, Ontario, Canada, L4B 1K7, (416) 764-5273. Note: Subscriptions are handled at this address ONLY. Subscriptions sent to our Buffalo address (above) will be forwarded to our Richmond Hill HQ. For best results, use the postage paid card at the centre of the magazine.

Quantity Orders: In Canada: Ingram Software Ltd., 141 Adesso Drive, Concord, Ontario, L4K 2W7, (416) 738-1700. In the USA: IPD (International Periodical Distributors), 11760-B Sorrento Valley Road, San Diego, California, 92121, (619) 481-5928; ask for Dave Buescher.

Editorial contributions are always welcome. Only original, previously unpublished material will be considered. Program listings and articles, including BITS submissions, of more than a few lines, should be provided on disk. Preferred format is Amiga 3.5-inch diskettes with ASCII text files. Manuscripts should be typewritten, double-spaced, with special characters or formats clearly marked. Photos should be glossy black and white prints. Illustrations should be on white paper with black ink only. IFF picture files or Aegis DrawPlus structured graphics files are preferred to hardcopy illustrations when possible.

All material accepted becomes the property of Croftward Publishing Inc., except by special arrangement. All material is copyright by Transactor Publishing Inc. Reproduction in any form without permission is in violation of applicable laws. Write to the Richmond Hill address for a writer's guide.

The opinions expressed in contributed articles are not necessarily those of Croftward Publishing Inc. Although accuracy is a major objective, Croftward Publishing Inc. cannot assume liability for errors in articles or programs. Programs listed in Transactor for the Amiga, and/or appearing on Transactor for the Amiga disks, are copyright by Croftward Publishing Inc. and may not be duplicated or distributed without permission.

Production
In-house with Amiga 2000 and Professional Page

Final output by Vellum Print & Graphic Services, Inc., Toronto

Printing
Printed in Canada by Bowne of Canada Inc.

Amiga Dispatches

Hard drive notes and help from Bryce

by Don Curtis

Don Curtis is a police officer in Denver, Colorado, who has for the last two years been assigned to program design and development along with system design and maintenance of ten AT & T Unix 3B2 computers. In his spare time, Don is an assistant sysop on Compuserve's AmigaForum.

I've been picking up lots of little bits and pieces off the networks lately and thought I'd use this issue's column to pass along as many as I can. By the way, I've tried as many of these items as I can, but I don't have all possible combinations of hardware and software, so can't guarantee each and every item. I can say that the information came from reliable sources, and where possible, has been verified by others trying it. However, as with all things, if you try something: *make sure you back up your data first*. That way, if things don't work quite as expected, you haven't lost anything except some time.

Hard Disk related information

Have you started using the new Fast Filing System (FFS) and suddenly discovered you're having re-validations occurring during a LoadWB from a startup script? That can occur if you've assigned T: to your Old File System (OFS) partition on the hard drive. Here's what's occurring. The OFS has a bug that causes an Inhibit to always succeed, even if you currently have a file opened for writing. When you Execute a script file, Execute writes to T: - when you do a LoadWB, Workbench checks everything on the device list to see if it's a disk drive or not. Well, Execute has your OFS T: directory open for writing and Workbench just inhibited it. So what does the system do... it re-validates the OFS partition to fix up the bit map. The fix is very simple: just assign T: to RAM: and the problem will go away.

Commodore has now provided a standard method for storing FFS on the hard drive itself in a loadable format. That means that, properly done, you can now Autoboot an FFS partition. Great Valley Peripherals can do it with their AutoBoot ROM as can the MicroBotics HardFrame. Commodore will be doing it in the near future. For those of you with access to the new v1.3 *include* files, you'll find the information contained in *devices/hardblocks.h* and *resources/filesysres.h*. The information is too complex to reproduce here but, simply put, it works. There's a major advantage to this also; you can update the file system without a ROM change.

Having problems getting a SCSI drive and/or peripheral working? It seems that some manufacturers of SCSI devices don't properly handle SCSI lines 20, 22, 24, 28, 30 and 34. They are supposed to be left unconnected, but some are tied to +5V and some are tied to ground. That can cause some obvious problems between devices if one manufacturer puts them on the 5V line and another ties them to ground and you try to hook up both devices to the same SCSI bus. There's a simple fix: just don't pass those lines in your SCSI interconnect cable.

Are you a bit miffed that your Amiga A2090 can't Autoboot even if you install the v1.3 OS ROM into your A2000? Commodore is coming to your rescue. No, they're not going to allow you to trade-in your A2090 for an A2090a, but what they *are* going to do is come out with a separate board that will allow Autoboot. The board will contain an Autoboot ROM that will then look for the A2090 and, if it's there, bring it on-line and boot from it. The board will take up an Amiga slot. No idea on cost or availability yet but it will require the Amiga to have the v1.3 OS ROM to work.

Michigan Software is also coming out with a similar device. Their board will have a battery backed RAM disk using static RAM. It will come with 64K of RAM and can be expanded to 256K using 32x8 static RAM chips. The RAM disk will be hardware write protected so it will be next to impossible to munge the data. It should be able to auto-mount all partitions and may be able to Autoboot from a FFS drive. It will also have user settable boot priority in case you have more than one Autoboot device on your system. Finally, it will have a switch-selectable Autoboot. That way, if you are still using the 1.2 ROM, your system won't crash. By the way, if you weren't aware of it, an Autoboot device will crash v1.2. This is why all current Autoboot devices (such as the A2090a) don't have the ROMs installed when delivered. Again, no information on when it will be available or what it will cost, only that it's in testing now.

The 1.3 manual says MaxTransfer should be specified in blocks; it should be specified in bytes...

If you've got an A2090a, you've probably read the warning about using Seagate SCSI drives (those with the 'N' suffix in the model name) as an Autoboot drive with it. A few bits of information on that issue. First of all, the problem only crops up on a *cold* start; it doesn't occur on a warm start. The reason is that the Seagate drives take longer to power-on and self-test than the board's time-out value. If you have a Seagate SCSI drive and an A2090a and want to Autoboot, there are a few things you can do. The first is separately power the Seagate and turn it on about 15 seconds before you turn on the Amiga. The second is to allow the cold boot to fail and then give the Amiga the three fingered salute (Ctrl-Amiga-Amiga). Finally, you can wait for Commodore to come out with their new driver; they are working on it and this will fix the problem also.

Janus drives under FFS

Do you want to use FFS on the Amiga partition of an MS-DOS side hard disk with the BridgeBoard? Yes, you can use FFS with JH0: (or JH1:, JH2:, etc.). Go through the normal low-level formatting and partitioning of the drive using FDISK and ADISK. Now, instead of using DJMount and DJFormat to mount and format the drive, use the normal AmigaDOS Mount and Format commands. The way to do this is to create a MountList entry for each JHx: partition you've created. The Device will be *jdisk.device* (and that file has to be in DEVS: or referred to by its complete pathname in the MountList). JH0: is Unit 0, JH1: is unit 1 and so forth. Use the proper start and ending cylinders as you set it up with FDISK and ADISK. Same goes for the number of heads, surfaces, etc. The rest of the information is the same as for any other Fast File System drive (Buffers, FileSystem, GlobVec, DosType, etc.). Use the sample MountList (with appropriate changes) on page A-3 of the 1.3 manual as a guide.

Once you've made sure you have it right, you can now mount and then format the drive using the normal AmigaDOS Mount and Format commands. You'll still have some slowdown due to the data going through the Janus interface, but you will get faster data access and an extra five percent of space on your drive.

Using the Xebec 9720HD

Have a Xebec 9720HD drive and want to use FFS? I've even got a possible fix for that setup. A warning is required here though: first of all, you *must* have Xebec's *scsi.device* v3.0. You can determine which version you have by going to the DEVS: directory of your boot disk and then from a CLI issuing the command **type scsi.device opt h**. Within the first few lines of the screen output will be a line telling you the version number. If you don't have v3.0, do *not* try running it as a FFS drive, you *will* lose data. Another warning is that not everyone who has tried this has had it work reliably. Most have, but some folks have reported that what seemed to work at first started giving them errors after a few days. This is probably related to an incorrect MaxTransfer value in the MountList, but that hasn't been verified as yet. Because of that, make frequent backups of your data in case a problem does crop up. Further on in this column I'll discuss MaxTransfer a bit more.

To set up the Xebec as a FFS drive, you simply forget about using their *scsimount* command and their *hdparms* file. You treat the Xebec just like any other Hard Drive and simply create a MountList entry for it using a typical FFS MountList. The Device would be *scsi.device* and the Unit number would be 1. If you've lost your documentation, the 9720HD has 4 heads, 612 cylinders (number 0 thru 611) and 17 sectors per track. Again, be sure you use the proper FileSystem entry, DosType and GlobVec. As above, look at the sample MountList on page A-3 of the 1.3 manual for guidance.

Setting MaxTransfer properly

Finally, are you using the MountList item MaxTransfer properly? MaxTransfer is a limiting parameter, not a speedup parameter. Some drive controllers can only handle limited amounts of data at once. If you request too much data at once, the controller will overflow and corrupt your data. The fix for this is the MaxTransfer parameter. MaxTransfer limits the amount of data that can be read or written to the drive in a single request to the number of *bytes* specified. The 1.3 manual says that the MaxTransfer should be specified in blocks, but that's incorrect; it should be specified in bytes.

A simple way to test to see if you need a MaxTransfer value in your MountList is to make a 'test1' and 'test2' directory on your hard disk. First, copy several large (100K filesize or more is good; the larger the file(s) the better the test) program files into the directory 'test1'. Then attempt to run the files. If they run okay, go to the next step. If they don't run okay, add a MaxTransfer parameter to your MountList. Start at 65536, re-boot and then try again. If they run okay now, go to the next step; otherwise, lower the MaxTransfer parameter by half to 32768, re-boot and try again.

Once the above works okay, copy the entire directory 'test1' into 'test2' at once using the ALL option of the copy command (**copy dh0:test1 to dh0:test2 all**). Now try running the programs again. If they all work, you're probably in good shape. If they don't, go through the above procedure to add a MaxTransfer parameter, re-boot and try again.

One other note on hard drives: Commodore has announced that all new Amiga models will come with a hard drive as

standard equipment. Not only that, but the hard drives will come with the system software already installed. A few days after that, they announced the Amiga 2000HD: An A2000 with A2090a controller and 40 Meg 28ms HD for $2,999. They also announced the A2500 which will be an A2000 with the A2090a, a 40 Meg 28ms hard drive and the A2620 68020 co-processor card with 2 Meg (expandable to 4 Meg) of 32-bit wide RAM for $4,699. By the way, the A2500 will be able to support Unix.

Video Toaster update

The Video Toaster is still coming from NewTek. The reason it isn't here yet is due to the RAM chip shortage and pricing. Right after it was announced, RAM prices jumped through the ceiling and NewTek said the Toaster would have to sell for $2500 for them to have been able to make a reasonable profit. That was *if* they could get the chips in the first place, and they couldn't get the quantities needed for production.

Because of that, NewTek decided to continue to work improving the Toaster rather than produce a trickle of them at outrageous prices. They have improved the video quality so that now it is 'dead-on broadcast spec RS-170A' and their software is more powerful. They have also re-engineered it to make it compatible with possible future add-on products. NewTek still won't give a release date, but feel that the chip situation is much better now and they are hoping to release it in the near future.

V1.3 information

Hopefully, you all know that v1.3 of the Amiga Operating System has been available for several months now. There have been a few interesting quirks and/or corrections that have popped up.

As noted above, on page 3-2 MaxTransfer is noted as the maximum number of *blocks* transferred. In fact, it limits the number of *bytes* transferred.

While the Format command shows the options FFS and NOFFS, in fact, Format reads the information in a MountList, if applicable, and will properly select the correct format method with or without the options. In fact, Andy Finkel says he's going to remove those options from Format in a future version.

If you have a 1 Meg Chip set (as some developers do) and use SetPatch, the manual is wrong on the 'R' option. It is case sensitive and *must* be lower case.

If you're using RAD: (The Recoverable RAM Disk) and FFS, RAD: has problems recovering. This can be corrected simply by adding a **Mount = 1** line in your MountList. If you do this, the RAD: device will recover as it should. One user reported it even recovered after several intervening re-boots without mounting the drive; on the final re-boot, he mounted it and it recovered fully.

If you want multiple RAD:s, you can have them. It requires a sector or file editor to make some modifications to *ramdrive.device*. First copy *ramdrive.device* to *ramdriv1.device* (you *must* keep the same number of characters in the filename!). Now edit that file using a sector or file editor (such as *NewZAP*). From 0, at byte offset 171 (hex) you'll find the ASCII character 'e'; change that to a '1'. If you view the file, you've just changed the string "ramdrive.device" to "ramdriv1.device". Now, go to byte offset 5a1 (again hex with byte 0 being the start of the file) and change the ASCII character '0' to '1'. You've just changed the string "RAMB0" to "RAMB1". Now duplicate your MountList entry for RAD: as RAD1: and change the "Device =" entry to *ramdriv1.device*. Mount RAD1: and you've got a second recoverable RAM drive. You can repeat this for a RAD2: or as many as you like and memory allows as long as you create a separate driver and MountList entry for each with the appropriate changes.

If you have a 1.3 ROM in your Amiga, you'll find that a few programs no longer work. Again, some developers have gone to hard-coded entry points in the 1.2 ROM and they changed under 1.3. I wish program authors would learn that there is only one absolute address you can depend on (AbsExecBase) in the Amiga. At least use that to find the correct entry points into the ROM.

In line with the above, a couple of short articles from Commodore-Amiga, written by Bryce Nesbitt , may be of interest:

You can have as many RAD:s as memory allows as long as you create a separate driver and MountList entry for each...

Disk Drives - what YOU are doing wrong!

by Bryce Nesbitt

Copyright 1988 Commodore-Amiga, Inc. Originally from AmigaMail™July/August 1988. Permission granted to reproduce provided this notice remains.

Several of our third party hardware and software developers have been using the floppy disk drives in an incorrect manner. Remember, using the *trackdisk.device* is always the safest procedure. If for some reason you go directly to the hardware, or build hardware, then this article is for you.

For software types

This part is primarily directed at people who write custom boot-loaders for game software. Due to defective loaders,

Programs that use simple timing loops will not work on all Amigas, and may fail on all new Amigas in the future...

there are thousands of Amiga owners who can't load some games, and will *never be able to buy them.*

The ultimate source for information on drive timing comes from the manufacturer's specifications. This article simply highlights the most misused points. You would be wise to obtain a copy of a vendor specification for a floppy disk drive. Most large electronics distributors would be happy to send one to you.

1) Don't make bad assumptions. For example; if you depend on the motor being on, turn it on before use. Don't assume that your boot code will be entered with the disk drive or system in any reliable state.

2) *Never* use a loop like this for timing:

```
            move.w   #$1000,D0
busy_wait   dbra     d0,busy_wait
```

This fails to produce accurate timing under a large number of circumstances. The speed of the above loop depends on what CPU is installed in the system, what video mode is selected, what type of memory the program is in, in what relation to vertical blank the code executes, what the blitter is doing, what interrupts are enabled and more other factors than you want to think about.

The 8520 chip provides fast, easy timing. See the companion article entitled "How To Waste Time".

3) The STEP line must be used as a low-going pulse. The direction must be set up *first,* with a separate write to the register. A typical use would be:

```
or.b     #%00000010,$bfd100   ;Set up direction
and.b    #%11111110,$bfd100   ;Pulse low
nop                           ;Wait a bit
nop                           ; "   "  "

or.b     #%00000001,$bfd100   ;Set it high again
;-- now wait 3 milliseconds for the head to
;-- get to the next track
```

We specify that our drives must get to the next track within 3 milliseconds. Some drives will step considerably faster, others will fail at or before 2.8 milliseconds. When the direction of step is changed, the settling time must also be added (a total minimum delay of 18 milliseconds).

Note that the TRACK ZERO sensor will not be valid until the head actually reaches the track.

4) When turning on the motor, wait for the READY signal to go low before reading or writing (steps are OK before then). Note that READY is only valid when the motor signal is ON.

5) To determine if a disk is in the drive, look at the DISKCHANGE signal. If it is low, the disk has been removed (and possibly inserted again) since the last check. Step the head to reset the latch and examine the current state.

6) Some code uses an extra track or two for storage or copy protection. We will not guarantee that our drives will have more than the normal 80 tracks. We will say that using one extra track is rather safe, two tracks is probably okay, and three tracks is a very bad idea.

7) After a disk write DMA has finished, a delay of 1.2 milliseconds is required before any other operations (drive select, step, head change, etc.). The type of disk drive we use has a gap between the erase head and the read-write head. The disk drive keeps the erase head enabled after the end of write gate to compensate for the gap. Failure to wait out the delay may result in writing over innocent data on other tracks or sides.

For hardware folks

1) The disk drive light should not flash on and off during access. Our drive activity light reflects the state of the motor. Typically the same signal is also connected to IN_USE (pin 4).

2) For compatibility with future systems, we require that drives refuse to step past track zero. That is, if the head is already at zero, and an outward step is received, it should not move the head. The drive must still reset its DISKCHANGE latch, however.

3) The critical specifications for our 90mm (3.5") drives are: 3ms track-to-track; 15ms settling time; >80% radial alignment using a Dysan Alignment disk; 500ms motor spinup; 800ms maximum power-on delay.

How To Waste Time

by Bryce Nesbitt

Copyright 1988 Commodore-Amiga, Inc. Originally from AmigaMail™ July/August 1988. Permission granted to reproduce provided this notice remains.

The worst way to insert a delay in an Amiga program is this:

```
        move.w #2000,d0
loop    dbra   loop,d0
```

Yet many custom disk-loaders use loops like this for step timing. This is *unacceptable*. Programs that use this method will not work on all Amigas, and may fail on *all* new Amigas in the future. Already there are cases of some programs that will not load on some machines.

If running under the multitasking operating system, the *timer.device* can provide a delay that still lets other tasks run. If taking over the machine, a loop like this is far better:

```
loop    btst.b #0,$bfed01
        beq.s  loop
```

This uses one of the high speed timer chips. As the following example shows, the timers are very easy to use. This loop is superior for a large number of reasons:

- This timer chip method is more accurate.
- The software loop method fails to produce accurate timing under a large number of circumstances. The speed depends on what CPU is installed in the system, what video mode is selected, what type of memory the program is in, in what relation to vertical blank the code executes, what the blitter is doing, what interrupts are enabled and more other factors than you want to think about.
- The timer chip can be "set and forgotten". The chip does the counting. Your code can continue to get other things done while the chip is counting.
- The timer can produce an interrupt when it is finished, or it can set a bit you can examine at any time.
- The timer can be set to automatically reload the count and start again. This gives even pulses, even if your software can't respond to them immediately.

Calculating the time

First some definitions:

1 millisecond (ms) = 1/1,000 second
1 microsecond (us) = 1/1,000,000 second
1 nanosecond (ns) = 1/1,000,000,000 second

On a stock 68000 based Amiga with no extra memory, the DBRA listed above will take about 1.5 microseconds per loop. The loop above wastes about (2000 * 1.5 = 3000) microseconds (This is the same as 3 milliseconds, or .003 seconds).

Each 8520 chip has two 16-bit timers counting down at .715909 Mhz, or 1.3968255 microseconds per tick. To get the same 3 milisecond delay with the 8520, we need to divide the desired time by the rate. 3000 / 1.3968255 = 2148.

A Complete Example:

```
;
; A complete 8520 timing example.  This blinks the power light at
; (exactly) 3 millisecond intervals.  It takes over the machine,
; so watch out!
;
;
; The base Amiga crytal frequencies are:
;          NTSC    28.318181 Mhz
;          PAL     28.37516  Mhz
;
;
;
; The two 16 bit timers on the 8520 chips each count down at 1/10
; the CPU clock, or .715909 Mhz.  That works out to 1.3968255
; microseconds per count.  Under PAL the countdown is a hair
; slower, .709379 Mhz.
;
; To wait 1/100 second would require waiting 10,000 microseconds.
; The timer register would be set to (10,000 / 1.3968255 =
; 7159).
;
; To wait 3 milliseconds would require waiting 3000 microsecsonds.
; The register would be set to (3000 / 1.3968255 = 2148).
;
; See the hardware manual for more information on the 8520 chips.
;
ciaatalo EQU $bfe401     ;Timer A low
ciaatahi EQU $bfe501     ;Timer A high
ciaaicr  EQU $bfed01     ;Interrupt control register
ciaacra  EQU $bfee01     ;Timer A control

  move.w  #$7fff,$dff09a      ;Kill all custom chip interrupts

;----Setup, only do once
;----This sets timer A to one-shot mode.
  move.b  ciaacra,d0          ;Set control register A on CIAA
  and.b   #%11000000,d0       ;Don't trash the 60/50Hz flag
  or.b    #%00001000,d0       ;or serial direction bits
  move.b  d0,ciaacra
  move.b  #%01111111,ciaaicr  ;Clear all 8520 interrupts

;----Set time (low byte THEN high byte)
;----And the low order with $ff
;----Shift the high order by 8
  move.b  #(2148&255),ciaatalo
  move.b  #(2148>>8),ciaatahi

;----Wait for the timer to count down
busy_wait:  btst.b  #0,ciaaicr          ;Wait for timer expired flag
  beq.s   busy_wait
  bchg.b  #1,$bfe001          ;Blink light
  bset.b  #0,ciaacra          ;Restart timer
  bra.s   busy_wait

  END
```

* * *

I hope the above information has been useful to you. As you can see, there are right and wrong ways to go about doing things. Let's hope that everyone chooses the correct methodology so that changes to the hardware or OS will no longer cause programs to 'break'.

Creative Computers

COMMODORE AMIGA™

SOFTWARE

B-PAINT	25.96
3-DEMON	71.95
4TH AND INCHES	29.24
64 EMULATOR 2, THE	49.95
AAARGH!	23.95
AC BASIC—COMPILER FOR AMI	134.06

Gridiron! ($9.95)

AC FORTRAN	199.00
ACCOUNTANT, THE	186.89
ACQUISITION 1.3	206.22
ADRUM	51.98
ADVENTURES OF SINBAD	32.46
AEGIS ANIMATOR	87.48
AEGIS ART PAK #1—CLIP ART	24.98
AEGIS DRAW	49.95
AEGIS IMAGES—PAINT	24.98
AESOP'S FABLES	31.23
AIRT SYMBOLIC LANGUAGE	44.95
ALIEN FIRES	24.98
ALL ABOUT AMERICA	37.47
ALOHA FONTS	12.96
ALOHA FONTS 2	12.96
ALOHA FONTS 3	12.96
ALTERNATE REALITY	27.06
AMEGAS	22.72
AMIGA DOS 1.3 WORKBENCH	24.95
AMIGA DOS EXPRESS	20.60
AMIGA FONTS (NTF)	59.95
AMIGA KARATE	24.98
AMIGA MACHINE LANG. DISK	11.95
AMIGA SYSTEM GUIDE DISK	14.95
AMIGA TIPS AND TRICKS DISK	11.95
ANALYTIC ART-GRAPHICS	37.48
ANALYZE 2.0—SPREADSHEET	93.73
ANIMAL KINGDOM	31.23
ANIMATE-3D	99.95
ANIMATION EFFECTS	32.46
ANIMATION MULTIPLANE	58.46
ANIMATION STAND	32.46
ANIMATOR'S APPRENTICE	184.38
ARAZOK'S TOMB	31.25
ARCADE ACTION PACK	34.95
ARENA	12.96
AREXX	32.95
ARKANOID	19.95
ART COMPANION	19.95
ART GALLERY FANTASY	23.36
ART GALLERY I	18.73
ART GALLERY II	18.73
ART OF CHESS, THE	22.95
ART PARTS #2	21.60
ASHA'S FONTS	58.95
ASSEMPRO	59.97
ATALK III	64.97
ATREDES (BBS)	97.46
AUDIO MASTER	37.48
AUSSIE JOKER POKER	32.46
AUTODUEL	27.00
AZTEC 68/AM-D	224.25
AZTEC C PROFESSIONAL	175.46
B.E.S.T. BUSINESS MGMT.	355.50
BAD CAT	30.25
BALANCE OF POWER	34.34
BARBARIAN	25.77
BARD'S TALE	36.00
BARD'S TALE CLUE BOOK	10.77
BARD'S TALE II	41.95
BARD'S TALE II CLUE BOOK	10.50
BASIC GRAMMAR SERIES	19.46
BBS-PC	62.32
BECKER TEXT	99.95
BENCHMARK C LIBRARY	64.97
BENCHMARK IFF LIBRARY	64.97
BENCHMARK MODULA-2	129.97
BENCHMARK SIMPLE LIBRARY	64.97
BETTER DEAD THAN ALIEN	21.95
BEYOND ZORK	33.76
BLACK CAULDRON	28.80
BLACK JACK ACADEMY	29.95
BLACK LAMP	22.00
BLACK SHADOW	22.71
BLITZKRIEG AT ARDENNES	34.41
BLOCKBUSTER	32.47
BOMB BUSTER	21.95
BREACH	25.95
BREACH SCENARIO DISK	16.21
BRIDGE 5.0	24.10
BRUSH WORKS	20.59
BRUSH WORKS 2	19.95
BUBBLE GHOST	22.71
BUTCHER 2.0	23.13
C-ZAR CZ-101 EDITOR	126.75
C.A.P.E. 68K ASSEMBLER	58.47
CAD PARTS AMIGA	16.50
CALLIGRAPHER	84.47
CAPITALIZATION SERIES	19.46
CAPONE	25.96
CAPTAIN BLOOD	32.46
CARRIER COMMAND	30.95
CASINO FEVER	25.96
CB TREE PLUS	64.95
CELEBRITY COOKBOOK, THE	22.71
CENTERFOLD SQUARES	19.95
CHESSMASTER 2000	32.40
CHICKEN LITTLE	19.48
CITY DESK	93.75
CITY DESK ART COMPANION	19.47
CLEVER & SMART	22.72
CLI MATE	24.98
CLIP ART #1	12.95
CLIP ART #2	12.95
CLIP ART #3	12.95
CLIP ART #4	12.95
CLIP ART #5	12.95
CLIP ART #6	12.95
CLIP ART #7	12.95
CLIP ART #8	12.95

SPECIAL
Aegis Promotion

Buy **VideoScape 3D** and get $10 off your purchase off the new **Modeler 3D** object design program or the NEW **Lights! Camera! Action!** animation presentation program.

• **COMIC SETTER**	**64.96**
COMIC SETTER (FUNNY DATA)	22.71
COMIC SETTER (SF DATA)	22.71
COMIC SETTER (SUPERHEROS)	22.71
CRAPS ACADEMY	25.95
CRAZY CARS	25.94
CROSSWORD CREATOR	34.34
CRYSTAL HAMMER	12.96
• **CUBEMASTER**	**14.95**
CUSTOMS SCREENS	43.73

GVP Great Valley Products

For Amiga 2000:	
IMPACT HC/20—20 MB Hard Card	549.00
IMPACT HC/45—48 MB 28ms Hard Card	799.00
IMPACT HC/40Q—42 MB 11ms Hard Card	799.00
IMPACT HC/80Q—84 MB 11ms Hard Card	1199.00
For Amiga 500:	
IMPACT A500-SCSI/HD20—20 MB Hard Drive	749.00
IMPACT A500-SCSI/HD45—45 MB Hard Drive	995.00
IMPACT A500-SCSI/HD40Q—42 MB 11ms High Speed Hard Drive	1049.00
IMPACT A500-SCSI/HD80Q—84 MB 11ms High Speed Hard Drive	1449.00
Quantum 80S, 84 MB 11ms 3.5" Hard Drive	995.00

Shock mounted, 64KB cache (for 11ms speed), SCSI interface. Compatible with IMPACT or A2090 boards.

Quantum 40S—42 MB version of above drive	599.00

• **CYGNUS ED PROFFESIONAL**	**64.97**
DARK CASTLE	25.95
• **DATA RETRIEVE**	**49.95**
DATA RETRIEVE. PROF.	211.60
DEATH SWORD	16.95
DECIMAL DUNGEON	31.23
DEEP SPACE	17.95
DEFCON 5	25.95
DEFENDER OF THE CROWN	34.34
DEJA VU	34.34
DELUXE HELP CALLIGRAPHER	22.71
DELUXE HELP FOR DIGIPAINT	21.84
DELUXE HELP FOR DPAINT II	21.84
DELUXE HELP FOR PHOTON	21.84
DELUXE MAPS	16.22
DELUXE MUSIC 2.0	69.95
DELUXE PAINT II	89.95
DELUXE PHOTO LAB	99.95
DELUXE PRODUCTIONS	139.95
DELUXE VIDEO 1.2	89.95
DEMONSTRATOR, THE	21.85
DES CARTES	22.71
DESIGN DISK, ARCH SCULPT	22.74
DESIGN DISK, ARCH VSCAPE	22.74
DESIGN DISK, FUTURE SCULPT	22.74
DESIGN DISK, FUTURE VSCAPE	22.74
DESIGN DISK, HUMAN SCULPT	22.74
DESIGN DISK, HUMAN VSCAPE	22.74
DESKTOP ARTIST	18.73
DESTROYER	25.26
DEVELOPERS TOOLKIT	36.22
DIGA-TELECOM PACKAGE	49.98
DIGI PIX #2	22.71
DIGI-DROID	69.95
DIGI-PAINT	41.22
DIGI-VIEW	143.72
DIGI-VIEW UPGRADE	13.95
DIRECTOR, THE	45.47
DISCOVERY EXPANSION DISKS	12.97
DISCOVERY/MATH	25.00
DISCOVERY/SPELL	25.00
DISCOVERY/TRIVIA	25.00
DISCRETE MATHEMATICS	36.22
DISK MECHANIC, THE	58.50
DISK TO DISK	34.34
DISK WICK	32.46
DISKMASTER	37.40
DOCTOR TERM PROFESSIONAL	74.06
DOMINOES	16.95
DOS TO DOS	37.82
DOUG'S MATH AQUARIUM	58.46
DOUGS COLOR COMMANDER	19.46
DPAINT ART&UTIL. DISK #1	21.60
DR. T'S 4-OP DLX EDITOR	98.95
DR. T'S BACH SONGBOOK	19.95
DR. T'S CZ RIDER	94.46
DR. T'S D-50 EDITOR	96.95
DR. T'S DRUMS	19.95
DR. T'S DX EDITOR	96.95
• **DR. T'S KCS**	**161.98**
DR. T'S KEYS	19.95
DR. T'S MATRIX	96.95
DR. T'S MT-32 EDITOR	97.50
DR. T'S KAWAI K-1 ED/LIB	92.97
• **DR.T'S MIDI RECORDING STUDIO**	**43.50**
DR. TS ESQ-APADE ESQ-1	96.95
DRAW PLUS (AEGIS)	162.48
DRUM STUDIO	32.47
DYNAMIC DRUMS	49.98
DYNAMIC STUDIO	142.96
DYNAMIC-CAD	299.00
EARL WEAVER BASEBALL	34.95
EBON STAR	25.96
EMERALD MINES	13.97
EMPIRE	34.32
ENLIGHTENMENT	15.95
EUROPEAN SCENERY DISK	17.95
EXCELLENCE	159.00
EXPLORER, THE	36.22
EXPRESS PAINT	62.50
FACC II	21.85
FAERY TALE ADVENTURE	31.23
FAERY TALE GUIDEBOOK	7.76
FANCY 3D FONTS	46.80
FANTAVISION	42.86
FERRARI FORMULA ONE	33.57
FEUD	12.96
FINAL ASSAULT	32.46
FINANCIAL COOKBOOK	14.40
FINANCIAL TIME MACHINE	31.16
FINE PRINT	41.95
FIRE N FORGET	25.96
FIREPOWER	15.60
FIRST LETTERS & WORDS	33.00
FIRST SHAPES	33.00
FLEET CHECK	25.96
FLIGHT PATH 737	16.21
FLIGHT SIMULATOR II	37.46
FLOW	62.32
FONTS AND BORDERS	22.72
FOOTBALL FACTS	42.97
FORMS IN FLIGHT II	79.50
• **FORMULA ONE AMIGA**	**19.46**
FOUNDATIONS WASTE	23.62
FRACTION ACTION	31.23
FROST BYTE	19.46
GALACTIC INVASION	16.22
GALAXY FIGHT	16.23
GALILEO 2.0	49.95
GALLERY 3-D (SCULPT)	51.96
GAME PLAY COMPACT DISK	12.95
GANYMED	21.95
GARRISON	29.19
GARRISON II	35.71
GEE BEE AIR RALLY	29.95
GEOMETRIC LIBRARY	11.88
GETTYSBURG	38.95
GIGANOID	12.96
GIZMOZ 2.0	39.95
GLOBAL COMMANDER	27.46
GOLD DISK FONT SET #1	21.85
GOLD SPELL II	29.95
GOLDEN PATH	29.21
GOLDRUNNER	24.98
GOMF	22.72
GRABBIT	20.59
GRAND SLAM TENNIS	31.25

Professional Page ($229)

GRAPHICS STUDIO, THE	38.96
GREAT STATES	24.99
GREAT STATES II	25.96
GRID START	16.22
GRID, THE	34.34
GRIDIRON-FOOTBALL GAME	9.95
GUILD OF THIEVES	30.90
GUNSHOOT	22.72
HACKER II	27.46
HAICALC	30.80
HARDBALL	28.12
HARRIER COMBAT SIMULATOR	32.95
HARRIER MISSION	16.21
HEAD COACH	32.47
HEX	24.95
HOLE IN ONE	25.99
HOLLYWOOD HIJINX	27.47
HOLLYWOOD POKER	25.96
HOME BUILDERS CAD	129.96
HOT & COOL JAZZ	21.60
HOT LICKS	27.59
HUNT FOR RED OCTOBER	27.46
• **HYBRIS**	**25.99**
ICE HOCKEY	32.49
IMPACT-BUSINESS GRAPHICS	62.46
IMPOSSIBLE MISSION II	32.46
INDOOR SPORTS	31.22
INSANITY FIGHT	25.96
INSTANT MUSIC 1.2	33.00
INTELLITYPE	35.17
• **INTERCEPTOR**	**37.95**
INTERCHANGE	29.22
INTERCHANGE FFII MODULE	16.95

WE CAN BEAT ANY ADVERTISED PRICE! BUT WE SELDOM HAVE TO — OUR PRICES ARE THE LOWEST! (IF YOU SEE A LOWER PRICE ON AN ITEM, GIVE US A CALL.)

Dynamic Studio ($139)

INTERCHANGE OBJECTS #1	16.95
INTO THE EAGLE'S NEST	28.56
INTROCAD	49.95
INVESTOR'S ADVANTAGE	64.97
• **INVISION**	**99.40**
IT'S ONLY ROCK & ROLL	21.60
J FORTH	68.74
JET	37.46
JET SET FONT SET	32.50
JINXTER	25.95
KAMPFGRUPPE	41.22
KARA FONTS	54.95
KARATE KID II	25.97
KARATE KING	15.56
KARTING GRAND PRIX	16.22
KEY TO C	22.72
KIDTALK	31.95
KINDERAMA	31.23
KINDWORDS	62.50
KING OF CHICAGO	34.34
KNIGHT ORC	30.90
KWIK SPEAK	28.04
LAND OF LEGEND	31.23
LARRIE	12.96
LATTICE C 5.0	CALL
LATTICE C PROFFESIONAL	CALL
LATTICE C++	357.46
LAZERSCRIPT	28.10
LEADER BOARD TORNA DISK	14.00
LEADER BOARD-GOLF GAME	27.00
LEARNING CURVE	51.96
LEARNING THE ALPHABET	19.46
LEATHERNECK	25.96
LEISURE SUIT LARRY	26.40
LEXCHECK	26.85
LIBYANS IN SPACE	19.97
LIFE CYCLES AMIGA	19.46
• **LIGHTS! CAMERA! ACTION!**	**59.95**
LINKWORD DUTCH	20.55
LINKWORD FRENCH	20.55
LINKWORD GERMAN	20.55
LINKWORD GREEK	20.55
LINKWORD ITALIAN	20.55
LINKWORD PORTUGUESE	20.55
LINKWORD RUSSIAN	20.55
LINKWORD SPANISH	20.55
LINT (GIMPLE)	63.70
LION'S AMIGA ART STUDIO	58.95
LITTLE DRAGON	12.96
LITTLE RED HEN	19.48
LOGIC WORKS	62.47
LOTTERY MAGIC	19.21
LURKING HORROR	25.96
MAD LIBS	12.48
MAGELLAN	119.00
MAGICAL MYTHS	32.47
MAGICIAN'S DUNGEON	21.84
MAJOR MOTION	25.96
MARAUDER II-DUPLICATION	27.47
MARBLE MADNESS	33.00
MASTERING CLI	25.96
MATCH IT	25.71
MATH MAGICIAN	27.49
MATH TALK	31.25
MATH TALK FRACTIONS	24.95
MATH WIZARD	31.23
MATH-AMATION	64.95
MAVIS BEACON TYPING	30.89
MAXIPLAN 500	93.13
MAXIPLAN PLUS	124.40
MEAN 18 COURSE DISK	14.96
MEAN 18 GOLF	28.77
MENACE	19.98
METACOMCO ASSEMBLER	68.72
METACOMCO PASCAL	68.72
METACOMCO SHELL	39.95
METACOMCO TOOLKIT	34.34
METASCOPE DEBUGGER	59.95
MICROFICHE FILER	69.95
MICROFICHE FILER PLUS	116.96
MICROLAWYER	37.47
MIDI MAGIC	97.47
MIKE THE MAGIC DRAGON	16.95
MIND WALKER	34.34
MIND LIGHT 7	153.95
MISSION ELEVATOR	34.95
• **MODELER 3-D**	**65.00**
MODULA-2 COMMERCIAL	206.22
MODULA-2 DEVELOPER	103.09
MODULA-2 REGULAR	68.72
MOEBIUS	39.95
MONEY MENTOR C VERSION	59.98
MULTI-FORTH	59.95
MULTI-PREFS	19.45
MUSIC MOUSE	51.35
MUSIC STUDENT	37.48
MUSIC STUDIO, THE	34.35
NEWSLETTER FONTS COLOR	29.95
NIMBUS 1:RECORD KEEPER	79.95
NINJA MISSION	13.00

OBLITERATOR	25.99
OFFSHORE WARRIOR	25.96
OGRE	32.46
ONE-ON-ONE	14.40
ONLINE 2.0	43.56
ORGANIZE	62.32
OTERA QUESTRON	25.96
P.O.W.	25.97
PAGE FLIPPER	31.23
PAGE FLIPPER PLUS F/X	103.97
PAGESETTER	93.72
PALADIN	25.95
PALADIN (QUEST DISK 1)	16.86
PAWN, THE	30.90
PCLO PRINTED CIRCUIT LAYOUT	199.95
PERFECT SCORE	54.97
PERSECUTORS	13.23
PHANTASIE	27.47
PHANTASIE 3	24.95
PHASAR-FIN'L MGMT	62.48
PHOTON PAINT	64.96
PHOTON PAINT EXPANSION DS	19.95
PHOTOSYNTHESIS	97.95
PINBALL I.Q.	19.46
PINK PANTHER	28.56
• **PIONEER PLAGUE**	**25.96**
PIXMATE	45.47
PLANET PROBE	19.46

Micron A2000 2 MB Board ($499

POLICE QUEST	29.97
PORTS OF CALL	29.71
POWERPLAYERS JOYSTICK	17.95
POWERWINDOWS 2.5	62.46
PRINTMASTER PLUS	31.23
PRO MIDI STUDIO	130.38
PRO VIDEO CGI	144.00
PRO VIDEO FONT SET #1	72.00
PRO VIDEO FONT SET #2	72.00
PRO VIDEO FONT SET #3	84.46
PRO VIDEO PLUS	184.95
• **PROFESSIONAL PAGE**	**229.00**
PROJECT D	31.23
PROSCRIPT	32.50
PROWRITE 2.0	78.10
PUBLISHER PLUS	124.98
PUNCTUATION SERIES	19.46
PUPPY LOVE	18.68
QUARTERBACK 2.0	45.4"
QUESTRON II	32.46
QUIZ MASTER	49.98
QUIZAM	23.10
R.R. AESOP'S FABLES	19.48
RAW COPY 1.3	38.97
• **REACH FOR THE STARS**	**27.46**
READ & RHYME	31.23

Excellence! ($159)

READ-A-RAMA	31.23
REALM OF THE WARLOCK	22.74
REASON: AT&T WRITERS W.B	271.56
RETURN TO ATLANTIS	34.32
ROAD RAIDERS	25.96
ROAD TO MOSCOW	29.97
ROADWAR 2000	27.47
ROADWAR EUROPA	29.21
ROADWARS	23.95
ROBBEARY	16.21
ROCKET RANGER	32.46

ROCKFORD	23.95
ROLOBASE PLUS	58.46
ROMANTIC ENCOUNTER	25.95
SAF T NET HD BACKUP	32.46
SANTA PARAVIA & FIUMACCIO	19.46
SARGON III	35.75
SCENERY DISK #7 EAST COAST	18.71
SCENERY DISK #11 EAST COAST	18.72
SCHEMATIC CAPTURE	199.95
SCRABBLE	27.46
SCRIBBLE	62.32
• **SCULPT 4D**	**441.96**
• **SCULPT 3D**	**69.95**
SDI: CINAMAWARE SERIES	34.34
SEASONS AND HOLIDAYS	21.60
SECONDS OUT	19.46
• **SEX VIXENS FROM SPACE**	**25.96**
SHADOW GATE	31.23
SHAKESPEARE 2.0	146.25
SHANGHAI-STRATEGY GAME	27.46
SHERLOCK	27.00
SILENT SERVICE	25.95
SILVER 3-D	109.95
SINBAD & FALCON	34.34
SKYFOX II	27.45
• **SLAYGON**	**19.95**
SMOOTH TALKER	33.95
SOCIAL STUDIES	19.46
SOFTWOOD FILE SG	78.21

• **SONIX (AEGIS)**	**49.98**
SOUND OASIS	64.95
TURBO PRINT	32.95
SOUND QUEST D-10 MASTER	97.50
SOUND QUEST D-110 MASTER	97.50
• **SOUND QUEST TEXTURE**	**93.60**
SOUND QUEST TX802 MASTER	113.75
• **SOUNDLAB**	**209.95**
SOUNDQUEST CZ MASTER	81.22
SOUNDQUEST D-50 MASTER	97.47
SOUNDQUEST DX7 MASTER	97.47
SOUNDQUEST DX7II MASTER	113.73
SOUNDQUEST GENERIC MASTER	81.22
SOUNDQUEST MT-32 MASTER	97.47
SOUNDQUEST SQ-80 MASTER	97.50
SOUNDQUEST TX81Z MASTER	81.25
SOUNDSCAPE UTILITIES 1	35.71
SOURCE LEVEL DEBUGGER	57.16
SPACE CUTTER	19.46
SPACE MATH	29.95
SPACE PORT	27.38
SPACE RANGER	13.00
SPACEQUEST	33.00
SPELLBOUND	25.96
SPELLER BEE	31.25
SPELLING IMPROVEMENT	19.95
STAR WARS EMPIRE STRIKES	30.25
STARFLEET I	29.95
STARGLIDER	29.50
• **STARGLIDER II**	**29.50**
STELLAR CONFLICT	25.95
STOCK MARKET-THE GAME	16.21
STREET CAT	12.47
STREET SPORTS BASKETBALL	17.95
STRIP POKER	27.46
STRIP POKER DATA #4	12.97
STRIP POKER DATA DISK #5	12.98
STUDIO FONTS V1 (COLOR)	29.95
STUDIO MAGIC	64.95
SUB BATTLE	24.95
SUMMER CHALLENGE	9.72
SUPER HUEY-COPTER GAME	23.36
SUPERBASE	93.73
• **SUPERBASE PERS. II**	**94.46**
• **SUPERBASE PROFESSIONAL**	**195.00**
SURGEON, THE	31.23
SYMPHONY SONGS (EACH VOL)	15.95
SYNTHIA	59.95
SYSTEMS MONITOR	28.95
T SHELL	32.46
TALES FROM ARABIAN NIGHTS	31.95
TALKING COLORING BOOK	18.73
TANGLE WOOD	25.96
TELEGAMES	23.95
TELEWARS	24.97
TERRORPODS	25.77
TEST DRIVE	34.95

OTHER CREATIVE CHRISTMAS PROMOTIONS:

- Framegrabber: $479!!! *Capture real time images in living color.*
- Spirit In 500 Amiga 500 RAM expansion board with 0K: $149!! *Expand your Amiga 500 with up to an additional 1.5 Megabytes of memory.*
- WordPerfect: The best non-graphic wordprocessor on the Amiga. $399 retail. Our price? ONLY $179!!!
- 1200 baud fully Hayes compatible modem with cables and software. $200 retail. Our price? $49.95!!!
- FREE VCR with purchase of computer—call for details.
- Package pricing on computers available—call for details.
- Amiga Live! 2000: $349
- Flicker Fixer! (A2000 only): $479
  Eliminate all flicker and scanline gaps 100%—optional multisync monitor.

NEW PRODUCTS

(As featured in this magazine)

VIDEO & AUDIO:
- Pro-Sound Designer
- MIDI Magic
- Digi View Gold
- Zoetrope
- Photon Cell Animator (In stock!)
- Sculpt-Animate 4D

GAMES:
- Operation Wolf
- Buble Bobble
- Rastan
- Dragon's Lair
- TV Sports Football
- Lords of the Rising Sun
- Cosmic Bouncer
- Rock Challenge
- Univeral Military Simulator
- Charon 5
- Joker Poker
- Advanced Dungeons & Dragons
- Final Assault
- Evil Garden
- Mission: Con-Bat
- Autodual
- Tower Toppler
- TeleEpic & TeleWar II

PRODUCTIVITY:
- Lattice C××
- Publishing Partner Professional
- Professional DataRetrieve
- ComicSetter
- Professional Draw
- Draw 2000
- Superbase Personal 2
- Superplan
- Deluxe Print II (Yes, in stock!)

ACCESSORIES:
- CMI MIDI 1
- INTRUDER ALERT (AMIGA ALARM)

PLEASE CALL FOR PRICES

TEXTCRAFT PLUS	64.93
THAI BOXING	14.26
THE EXPLORER	32.47
THEXDER	23.95
THREE LITTLE PIGS	19.48
THREE STOOGES	35.95
THUNDERBOY	22.71
TIME BANDITS	22.09
TOOL CADDY	32.46
TRANSCONTINENTAL RAILROAD	25.96
TRANSPORT CONTROLLER PHOTON	194.97
TRIGONOMETRY	36.22
TRUE BASIC	68.72
TURBO	16.95
TURBO SILVER 3.0	129.96
TV SHOW	65.01

AmiGen ($149)

TV TEXT	62.32
TXED PLUS	51.95
ULTIMA III	28.57
ULTRA DOS	37.95
UNCLE D CON SOUND TRATION	25.97
UNINVITED	34.34
VAMPIRE'S EMPIRE	29.21
• **VIDEO EFFECTS 3D**	**129.96**
VIDEO VEGAS	24.10
• **VIDEOSCAPE 3D**	**124.98**
• **VIDEOTITLER**	**99.95**
VIP PROFESSIONAL	99.00
VIP VIRUS	32.46
VIRUS (GAME)	19.46
VOCABULARY IMPROVEMENT	19.95
W SHELL	31.16
WARLOCK	22.71
WBEXTRAS	24.95
WESTERN EUROPE (FLIGHT SIM)	18.16
WESTERN GAMES	31.95
WINDOW PRINT II	22.75
WINNIE THE POOH	16.50
WINTER CHALLENGE	9.95
WINTER GAMES	27.46
WORD MASTER	29.95
• **WORD PERFECT**	**179.00**
WORD PERFECT LIBRARY	84.50
WORKS, THE	124.97
• **WORLD CLASS LEADER BOARD**	**25.97**
WORLD GAMES	27.46
WORLD TOUR GOLF	27.45
WRITE 'N FILE	59.95
WSHELL	32.47
• **X-CAD**	**389.00**
XXX RATED GRAPHICS	19.95
• **ZOETROPE**	**89.95**
ZOOM	21.95
ZORK TRILOGY	48.10
ZUMA FONTS VOL 1	21.85
ZUMA FONTS VOL 1 THRU 4	65.85
ZUMA FONTS VOL 2	21.85
ZUMA FONTS VOL 3	21.85

HARDWARE

8-UP MEMORYCARD 0K	145.00
ALEGRA WITH 0K	166.95
AMIGA DOS 1.3 ROM	39.95
AMIGA DOS 1.3 ROM + INST.	59.95
AMIGA LIVE!	270.00
AMIGA LIVE! 500	289.00
AMIGA LIVE! 2000	349.00
AMIGEN GENLOCK	149.00

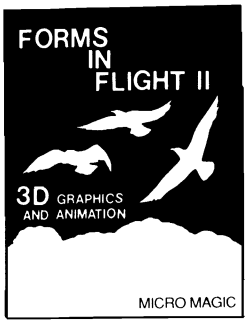
ASDG 8 MEG BOARDS W/0K	299.00
BYTE BOX 0K-RAM OPTIONAL	249.00
• **CA-880 FLOPPY DRIVE**	**159.00**
COLOR SPLITTER	79.95
CPS 500-POWER SUPPLY A500	74.97
EASYL 1000 TABLET	369.00
ECE MIDI 500/2000	48.71
ESCORT 2 UNPOPULATED	249.00
ESCORT 2 WITH 1M	569.00
ESCORT 500 UNPOPULATED	299.00
ESCORT 500 WITH 1M	619.00
EXP-1000 1M POPULATED FOR A500	479.95
EXP-1000 1M UNPOPULATED FOR A500	219.95
• **FLICKER FIXER (HARDWARE)**	**479.00**
• **FRAMEGRABBER**	**479.00**
FUTURE SOUND—AUDIOSAMPLER	142.20
• **GEN ONE GENLOCK**	**739.00**
HARD DISK CONTROLLER	319.00
HARDFRAME SCSI CARD	239.00
IMPRINT	399.00
LASER JET BLACK TONER	104.00
MAGNI 4004 GENLOCK	1595.00
• **MICRON 2 MEG FOR A2000**	**499.00**
• **MICRON 2 MEG FOR A500**	**599.00**
MIDI GOLD 1000 OR 2000	64.20
MINISCRIBE 20MB 3.5" FAST	299.00
MINISCRIBE 8051S SCSI 40M	599.00
NEC P5200 PRINTER	CALL
OKIMATE 20/PLUG N PRINT	249.00
OVERDRIVE HD CONTROLLER	199.95
PAINT B/W CARTRIDGE (HP)	23.60
PAINT COLOR CARTRIDGE (HP)	29.10
PAINT CUT SHEET (HP)	19.10
PAINT TRANSPARENCY (HP)	54.05
PAINT ZFOLD PAPER (HP)	14.95
• **PANASONIC WV1410 CAMERA**	**224.96**
PANASONIC WV1500 CAMERA	319.95
PERFECT SOUND DIGITIZER	67.47
• **PERFECT VISION**	**169.95**
PRO GEN GENLOCK	379.00
PRO RAM 8MB 0K BOARD	249.00
QUANTUM PRODRIVE 80S	995.00
QUEST FOR CLUES (BOOK)	18.95
RAM EXPAN./CLOCK 0K	30.00
SCRIBE-CARD 30 FOR 2088D	399.00
SOUND SAMPLER	86.63
SPIRIT 0 MB FOR A1000	249.00
SPIRIT 0 MB FOR A500	149.00
SPIRIT 0.5 MB FOR A1000	361.00
SPIRIT 0.5 MB FOR A500	309.00
SPIRIT 1.0 MB FOR A1000	473.00

Magellan ($119)

SPIRIT 1.0 MB FOR A500	470.00
SPIRIT 1.5 MB FOR A1000	585.00
SPIRIT 1.5 MB FOR A500	630.00
STAR NX1000 PRINTER	199.00
STAR NX1000 RAINBOW	279.95
STARBOARD 2 SCSI MODULE	101.36
STARBOARD 2 UPPER DECK	79.95
STARBOARD 2/A1000 0K	299.00
STARBOARD 2/A500 0K	299.00
STARBOARD SB2000 ADAPTER	42.00
STEREO SOUND SAMPLER STRV	87.75
SUBSYSTEM 500	199.95
SUPERGEN	699.00
• **SUPRA 2400 MODEM**	**149.00**

WE CAN BEAT ANY ADVERTISED PRICE! BUT WE SELDOM HAVE TO — OUR PRICES ARE THE LOWEST! (IF YOU SEE A LOWER PRICE ON AN ITEM, GIVE US A CALL.)

SUPRA DRIVE 20 MEG A1000	699.00
SUPRA DRIVE 20 MEG A500	699.00
SUPRA DRIVE 30 MEG A1000	799.00
SUPRA DRIVE 30 MEG A500	799.00
VI 2000	69.95
VI 2000 RF	79.95
XEROX 4020 INK JET COLOR	1140.00
XEROX 4020 STARTER KIT	156.00



Forms In Flight II ($79.50)

ACCESSORIES

A1084 CARRING CASE	51.94
A500 CARRING BAG	60.20
ALPS 224E COLOR PRINTER	699.00
AMIGA 2000 DUST COVER	10.79
AMIGA 500 DUST COVER	6.97
CABLE, DB25M-DB25F 8 FT.	16.95
CABLE, DRIVE EXT.	29.95
CABLE, DRIVE EXT. 2 FT.	19.95
CABLE, FLCK FIX TO NEC II	18.95
CABLE, FLCK FIX-SONY 1302	18.95
CABLE, IMAGEWRITER A1000	21.42
CABLE, IMAGEWRITER A2000	21.42
CABLE, PARALLEL A1000	19.00
CABLE, SERIAL A1000	19.00
CABLE, SERIAL A500/A2000	19.00
CABLE, PARALLEL A500/A2000	19.00
CENTRONICS-36 2WAY BOX	35.00
CERTRON 80 DISK HOLDER	14.95
CLEANING KIT (ALLSOP)	14.95
CLEANING KIT (SMALL)	8.95
COPY STAND, 2 LIGHTS, STD	64.46
CURTIS DIAMOND	34.03
CURTIS DIAMOND PLUS	46.42
CURTIS EMERALD	40.22
CURTIS PHONE LINE SURGE	12.36
CURTIS RUBY	58.81
CURTIS SAPPHIRE	57.04
DELUXE MIDI INTERFACE	59.96
DESK JET BLACK INK	15.78
DESK JET DUST COVER	29.17
DISK FILE 30	7.95
DISK FILE/10	4.57
DISK FILE/30 W/LOCK	8.95
DISK FILE/5	2.30
DRIVE CLEANING KIT	6.95
EASYL 2000 TABLET	369.00
EASYL 500 TABLET	369.00
EPYX 500XJ JOYSTICK	13.69
FLICKER MASTER	13.95
FOUR PLAYER ADAPTER	11.95
GENDER CHANGER/ BENDER	20.58
GRAVIS MARK VI JOYSTICK	28.56
GUIDE TO THE AMIGA	20.95
IMPACT 0M/2M SCSI	299.00
IMPACT SCSI 0M/1M	299.00
JOYSTICK Y ADAPTER	10.95

NEW PRODUCTS ARRIVE DAILY. CALL FOR UNLISTED ITEMS.

AMIGA 500 MONITOR STAND	29.95
AMIGA BAG	29.95
AMIGA BRASS KEY RING	3.50
AMIGA COFFEE CUP	5.50
AMIGA DISK HOLDER	12.50
AMIGA FASHION GOLF SHIRT	19.95
AMIGA FASHION JACKET	39.50
AMIGA SLEEVE TSHIRT	7.95
AMIGA TERRY CLOTH VISOR	5.50
AMIGA TOWEL	15.75
AMIGA TWILL CAP W/CORD	7.95
• AMIGA LIGHT PEN (NEW)	97.47
APRODRAW 12X12	499.59
C-VIEW A500 COLOR COMPOS.	36.19
CABLE, 15' CAMERA	12.97
CABLE, AMIGA TO NEC II	24.95
CABLE, AMIGA TO SONY 1302	18.95
CABLE, AMIGEN TO 1084	26.00
CABLE, CENTR. TO CENTR.	15.95

Boing! Mouse

Professional Optical 3-button mouse for your Amiga. Available EXCLUSIVELY at Creative Computers for Christmas 1988!

The first and only Optical Mouse for the Amiga $114

This mouse is definitely the best mouse you can buy for your Amiga.

- **Optical technology eliminates friction and momentum coefficients for super-precise handling.**
- **No moving parts for increased reliability.**
- **Middle button makes it the only mouse compatible with A2024 software and X Windows.**
- **Compatible with ALL Amiga computers.**

Available ONLY at Creative Computers for this Christmas season.

KAO BLANK DISKS	17.95
KRAFT JOYSTICK	8.54
LABTEC SPEAKERS	99.95
LENS 16MM FOR WV1410	21.85
LIGHT GUN	32.46
LIPSTICK+ VOICE CONTROL	26.97
MAZEMASTER JOYSTICK	8.54
MC68881 CRYSTAL OSCILLAT	5.00
MC68881 RC161 MATH CO.	201.95
MD 60L WOOD DISK HOLDER/L	26.21
MD 64 WOOD DISK HOLDER/60	22.46
MICHTRON DISK FOLDER	10.50
MICRO 50 DISK HOLDER	6.50
MICROSTORE DISK HOLDER	9.71
MIDI LINK 6' MIDI CABLE	7.30
MIMETICS MIDI INTERFACE	43.00
MONITER EXT. CABLE 4'	21.95
MONITOR STAND (15")	59.98
MOUSE EXTENSION CABLE	9.95
MOUSE HOUSE (MAX)	9.95
MOUSE HOUSE (MILLIE)	9.95
MOUSE MATS	6.95
MOUSE TOP	9.95
MOUSE WASHERS	5.95
NEC P6 RIBBON	16.16
NX1000 DUST COVER	6.97
OKIMATE PLUG N PRINT	79.95
PAINT B/W CARTRIDGE (HP)	23.60
PAINT COLOR CARTRIDGE (HP)	29.10
PAINT CUT SHEET (HP)	19.10
PAINT TRANSPARENCY (HP)	54.05
PAINT ZFOLD PAPER (HP)	14.95
POWERBUSS	13.00
PRINT & WEAR	13.95
PRINTER CABLE A1000	19.00
PRINTER CABLE A500/A2000	19.00
PRINTER PAPER 1000 COUNT	17.95
PROCESSOR ACCELERATOR CMI	159.00
QUICKSHOT II JOYSTICK	10.95
SK080 DISK FILE 80 3.5"	19.95
SONY DISKS 10 HARD PACK	21.95
SONY DISKS 10 PACK	20.95
SONY MONITOR CABLE	27.30
SPEAKERS	64.96
SQUARE AD PAD	4.99
SUNCOM PRINTER STATION	4.49
SUPER THREE-WAY JOYSTICK	24.25
TEAK-TECH 120 DISKHOLDER	29.95
TEAK-TECH 60 DISKHOLDER	27.95
TEAK-TECH 64 DISKHOLDER	24.95
TIMESAVER	57.97
TOUCH IT STATIC WRAP	6.95
TRACKBALL (MOUSE EMULATION)	39.95
TRANSVESTOR 1000	23.36
TRANSVESTOR 2500	41.86
UNIDRIVES 3.5	159.00
VI 500 INTERFACE	58.50
VI 500 RF MODULATOR	69.60
WAVE PAD	7.97
• X-SPECS 3D	99.95

XEROX 4020 4-PACK INK	19.95
XEROX 4020 BLUE INK (2)	9.95
XEROX 4020 RED INK (2)	9.95
XEROX 4020 YELLOW INK (2)	9.95

BOOKS

AMIGA ASSEM LANG BOOK	11.50
AMIGA BASIC INSIDE & OUT	19.46
AMIGA DEV. REF. GUIDE	14.95
AMIGA DISK DRIVES: IN-OUT	25.15
AMIGA DOS MANUAL	22.95
AMIGA MACHINE LANGUAGE	18.95
AMIGA PROG. HANDBOOK I	20.28
AMIGA PROG. HANDBOOK II	20.28
AMIGA TRICKS AND TIPS	18.95
AMIGAZINE	7.95
AMNEWS	14.95
DESKTOP VIDEO BOOK	12.95
FAST ACCESS WORD PERFECT	12.95
INSIDE AMIGA GRAPHICS	16.16
INSIDE AMIGA WITH C	22.46
KICKSTART GUIDE (BOOK)	19.50
MASTERING AMIGA DOS—BOOK	11.47
SENTRY MAGAZINE	3.00

Creative Computers is both a mail order company with a store's support and three store showrooms with mail order prices. Visit one of our showrooms today. You'll be amazed!

RETAIL SHOWROOM LOCATIONS

SANTA MONICA & WESTSIDE—318 Wilshire Blvd., Santa Monica, CA 90401. Tues.-Sat. 11-7 p.m./Sun. 11-5 p.m./Phone: 213-394-7779

LAWNDALE & SOUTH BAY—4453 Redondo Beach Blvd., Lawndale, CA 90260. Mon.-Sat. 11-7 p.m./Phone: 213-542-2292

VENTURA, OXNARD & SANTA BARBARA—2112 E. Thompson Dr., Ventura, CA 93001. Tues.-Sat. 11-7 p.m./Sun. 12-5 p.m./Phone: 805-652-0325

The Creative Computers Advantage:

- Authorized Amiga Dealer
- The largest dealer of Amiga products in the World
- Three store locations means excellent support
- Amiga-specific—unlike the competition, we don't claim being Amiga specific while selling other brands under another business name
- Authorized service center
- Uniform low pricing and largest selection, no hidden costs or catches
- We don't charge your card until the product ships

Customer Service: Call 213-542-2292 from 11 a.m. to 7 p.m. PST for technical questions.

VISA MasterCard AMERICAN EXPRESS Card DISCOVER

***Shipping Info:** No surcharge for Visa/MasterCard; 4% surcharge for American Express; 1% surcharge for Discover. Shipping within the continental United States for software orders over $100 (U.S.) is FREE. Please call for all other shipping information. Educational, corporate and aerospace purchase orders accepted.

Return Policy: Defective merchandise under warranty will be repaired or replaced. Returned products must be in original condition and packaging. No refund on defective products or products that do not perform satisfactorily. We make no guarantees for product performance.

Conditions: Creative Computers reserves the right to limit the sales of any items to local in-person pick-up only. Prices and availability subject to change without notice.

Call Our 24-hour BBS: Call 213-394-5988 (using your modem). Or, visit one of our stores today!

Creative Computers Goes International!

Toll-Free Numbers: Minimum orders $100 (U.S.). Orders only please. Absolutely no product inquiries or any questions will be taken on these lines.

International Phone Numbers: Canada 1-800-548-2512; Italy 1678-74086; United Kingdom 0800-89-1178; France 19-0590-1099; W. Germany 01-30-810466; Japan 0031-11-1351; Australia 0014-800-125-712; Netherlands 06-022-8613; Norway 050-12029; Denmark 0434-0297; Switzerland 046-05-3420.

International Ordering Policies: We have operators versed in the following languages: English, French, and Spanish, only. Shipping prices vary and are quoted following your order. Payment is made via credit card or electronic fund transfer. For any other information please call or regular toll lines. Absolutely no shipping inquiries will be accepted over the toll-free phone lines.

Creative Computers

4453 Redondo Beach Blvd., Lawndale, CA 90260 Mon.-Sat. 8 a.m.-6 p.m. PST **FAX:** 213-214-0932

ORDERS ONLY: 800-872-8882 OUTSIDE CA **213-370-2009** INSIDE CA

WE CAN BEAT ANY ADVERTISED PRICE! BUT WE SELDOM HAVE TO — OUR PRICES ARE THE LOWEST! (IF YOU SEE A LOWER PRICE ON AN ITEM, GIVE US A CALL.)

ViewPort

Will Workbench 1.4 do it all?

by Larry Phillips

Larry Phillips is an Amiga Hardware and software hacker from Vancouver, British Columbia, and a Sysop on Compuserve's AmigaForum.

As might have been predicted, Commodore's release of 1.3 has stirred a lot of discussion about 1.4, and much of that discussion centres around one of the primary aims of 1.4, the improvement of the WorkBench environment. The Amiga is a machine of choices, in everything from colours and pointers to hardware addons, and most visibly, in choice of user interface; CLI or WorkBench.

Unfortunately, only one of the choices of user interface can be used for all things, while the other is of limited use. The CLI can be used for everything, and in fact, the WorkBench need never be loaded in order to make full use of the machine. The WorkBench, however, is less than adequate when compared to the CLI for anything beyond the running of applications and the most rudimentary routine 'maintenance'. This state of affairs has prompted the writing of many tools that attempt to make the WorkBench more powerful. HandyWB, Browser, IconX, and so on, are all trying to overcome shortcomings in the WorkBench environment. They do a pretty good job in the areas they are aimed at, but the WorkBench problems go deeper than that. Unfortunately, the solutions put forth for a general WorkBench facelift do not always address the real issues, but merely scratch the surface. I'd like to address some of the proposed solutions in this light, and point out a few of the problems inherent in them.

The WorkBench, as it now stands, is 'single-threaded'. Not to be confused with single-tasking, this simply means that you must finish certain operations that you start, before you can start another. Have you ever been annoyed by the 'sleeping cloud' while waiting for something to finish loading? You click on a window, it opens immediately, and the disk starts looking for the icons. There's the program you wanted; the first icon that appears. Frustration sets in, because there are 5 or 6 more icons that need to be found and displayed before you can actually click on the one you want. You curse the WorkBench. "This is a multitasking machine," you say, "why can't I just keep clicking on things?"

I wish the answer were simple. It boils down to the WorkBench not really knowing what you want to do. Are you telling it to stop what it is doing and start the next operation? Perhaps you want it to start the next operation while continuing with the first one? Perhaps what you really mean is 'start the next operation when this one is finished'. If the first case is assumed, you can be assured that there would be many times when you really needed to see the rest of the icons, which of course would never appear because the icon gathering had been stopped, and of course aborting a program to load another is probably not what you want at all, so different operations would have to be treated in different ways.

The second choice, starting the second operation while the first is still working, is perhaps the worst of all. It would be especially annoying to those using only floppy disks, because the two operations would be in contention for disk resources. If you have never loaded two programs simultaneously from a floppy disk, try it someday using two separate CLIs. They will fight for head position, and though they will both eventually load, it is a long and noisy process. Given all the complaints about the slow directories and program loading from floppy disk, allowing competition for disk resources from the WorkBench would do the Amiga more harm than good.

The last choice, allowing you to 'click-ahead' is relatively harmless. Much like 'type-ahead', it will do the job, though it won't really speed anything up, but does give a user a better feeling about the environment.

Another possibility, which I have yet to see mentioned, is that the operation in progress is suspended while the next starts up, and continues when the next one is done. This presents its own

"...Is there really any difference between seeing all files on the Workbench and using a CLI?.."

problems, especially on a 512K machine, unless care is taken to prevent running out of memory from multiple tasks being held in a suspended state as you launch more and more programs.

Another area being heavily discussed is that of the 'visibility' of files. A CLI allows you to see every file in a directory, which is one of the main reasons that the CLI is more suited to full control of the machine. The WorkBench, on the other hand, allows you to see only icons, which has far reaching ramifications, both in usefulness of WorkBench, and in the perception of the machine by a user.

The icons, of course, can be looked at as representations of objects. You click (or double click) on an object, and expect something to happen, exactly what happens being dependent on the type of object you are clicking on. This is fine for such things as launching programs or opening drawers, but there are often more operations you might want to perform using that object, and you are hampered by the iconic representation. Since it does represent an object that does a particular thing, there must be another method, over and above clicking the icon, to indicate to the machine that you want to do something different from the normal operation. This has been addressed, partially, by menu selections on the WorkBench for such things as Duplicate, Rename, Info, Discard, etc.

The main problem here is that the icon, while representing an object, does not necessarily represent the right sort of object for another type of operation. Copying a program to another disk is a good example. When we click and hold on an icon, and drag the image to another disk or drawer, we are doing more than copying the icon itself. We are copying the icon and the program of the same name. Unfortunately, many programs are not fuly contained in one file, and perhaps not even in one drawer. The object has changed, in effect, for the copy operation. What we really want to do is to copy the entire program, including any and all support files.

The most common solution put forward for this problem is that we need to be able to see all the files on any given disk by using the WorkBench. Some say we should see icons for those files that have icons, and text for those files that have no icons, while others argue for 'default image' icons to fill in for files that have no icon specified. This solution seems, at first glance, to be simple and straightforward, handling all cases well, but is (as usual) more complex. The complexity stems from the philosophy of the user interface itself. WorkBench is aimed at making things easy for the user, and particularly for the new computer user. It is supposed to remove the intimidation of computers, hiding the gory details, and letting the user look at things in terms of jobs to be done, rather than at the details of how those same jobs have to be set up to be run.

It is all too easy for an experienced Amiga owner to say that WorkBench should allow one to see all files, without considering the effect on the new user of a solid wall of icons, or a screen full of file names, perhaps extending off-screen, requiring scroll bars or some other means to see them all. This loses sight of the reason for the WorkBench environment in the first place. Imagine that first-time user, seeing fonts, device drivers, libraries, and so on, clicking on them, and wondering why they didn't do anything useful. Is there really any difference between seeing all files on the WorkBench and using a CLI?

My feeling about the WorkBench is that yes, we need to be able to do a lot more with it, and yes, we need to be able to see all files with it, but that these needs must never lose sight of the underlying philosophy of the mouse/icon interface. There are many ways to do this, and some of them would require very rework of WorkBench, while others would require that old methods be retained in addtion with newer methods, in order to ensure that older programs still worked properly.

The WorkBench should probably default to being much like it is now, with only those icons showing up that make sense in terms of jobs to be done, retaining the flavour of the application-oriented nature of the WorkBench. Further options could expose more of the underlying structure of the files on a disk for purposes of copying entire applications to another disk or to manipulate all files 'belonging' to one application. Another option might expose more files, perhaps all files, to give the user an ultimate, CLI-like control over the machine.

Many good ideas have been put forward by Amiga owners. User definable menus, backdrop patterns, 'gravity' for positioning disk icons along one edge of the display, proper aspect ratio for icons on interlaced screens, different methods for extended selection, etc., are all good ideas, and many have no negative impact on the new or experienced user. Some of these ideas can be considered nothing more than cosmetic, and are important in their own way. More important though, is that all users should be comfortable with the user interface that suits them best, and the only way to ensure that is to make the CLI and WorkBench environments equally powerful. Only by providing choices without severe compromises can we expect the Amiga to appeal to everyone.

The Mac suffered for years with Apple's attempts to restrict the machine to mouse/icon usage only, and the MSDos machines suffered from lack of anything other than the command line interface. Later, third party efforts tried to rectify the shortcomings of each of these. The Amiga is the first machine that made an attempt to offer the best of both worlds right out of the box, and it did a pretty fair job of it. It certainly isn't perfect, especially in the WorkBench area, and it could be improved considerably, provided that the basic philosphies of each user interface are always taken into account.

ACCESS

The world of non-commercial software

by Steve Ahlstrom
Copyright (c) 1988 Steven Ahlstrom

As the primary Sysop on CompuServe's Amiga forums, and a veteran Amiga programmer, Steve Ahlstrom sees just about every notable public domain and shareware program that gets written, and in this column he tells you about the ones that have recently caught his eye. If you have some information you'd like to share through this column, you can leave a message to Steve on CompuServe (his PPN is 76703,2006), or send the info to us and we'll pass it on.

Disk Storage 5.0
Freely distributable software by Khalid Aldoseri

Have you ever been merrily working away, then tried to save your data only to find out that you haven't any free space on your disk? Wouldn't it be nice if there was a program you could run from your startup-sequence to monitor such things and let you know before you get in a bind? Thanks to Khalid Aldoseri, there is, and it's called Disk Storage (DST).

DST probably gives you more information about your disk drives and RAM than you ever wanted to know but, when that need arises, it sure is nice to have it in your system.

When you run it, DST will open a window giving you information like the following:

Name:	Size	Free	Alarm
DF0:	837	10	50
DF1:	837	129	50
DF2:	- No Disk Present -		
DH0:	19864	6742	993
DH1:	19734	1118	986

This tells you, for instance, that on DF0:, of 837 kilobytes available, only 10K is free for your use. The alarm feature is user settable for any given number. In this case, you would be informed when the free space on DF0: has dropped below 50K free. DST will pop up on any screen so you won't get the warning and not know about it.

As well as being able to set the alarm level to any number you wish, you may also set what type of alarm you want. You may set it up to flash all screens five times while also flashing the relevant item in DST's window, to bring up a requester (Workbench will be brought to front in this case), have the Narrator kick in and tell you about the problem, or ignore the warning other than flashing the entry in DST's window. The default warning level is five per cent of the total capacity of the device, with a minimum alarm level of 50 kilobytes.

DST's window will be as large as it needs to be to display all mounted devices. There is a limit, however, and that limit is 20 devices.

Various other bits of information are given in DST's window display. If a floppy disk has its write protect tab in place, an asterisk will appear next to the 'Free' number corresponding to that disk. If the disk is being validated by AmigaDOS, a 'V' will be shown.

DST automatically updates every three seconds, so you can run it and forget about it. Even though three seconds is a relatively small amount of time, there are those who are more anxious than others; to accommodate them, DST will update immediately if the user clicks on its 'titles' line. You may also change its update time if you choose.

DST offers many user definable options. You may remove devices from the list, remove just floppies from the list, switch memory display on and off, switch time display on and off, put the program to sleep, and more.

Did I say switch time display on and off? Yes, you can have the date and time displayed in DST's window title area. You can also have DST display RAM usage. Like many other utilities, it will display available chip RAM, available fast RAM, and total available RAM. This in itself is not an unusual feature. However, DST extends this feature in an unusual and useful way: you can have it display 'memory chunks' rather than total available memory. This will diplay the number of free memory blocks currently in your system. This can be very handy to know, as it will give you an idea of just how fragmented your memory is.

There a lot more features to this program and I use them all. This is one of those little programs that finds itself run from my startup-sequence; I usually just forget about until it needs to warn me of a storage problem. DST occupies about 18K in memory and consumes less than half a per cent of your processor time (at an update rate of five seconds).

From the documentation: "This program is placed in the Public Domain, you can spread it as much as you like as long as you keep this file with it, and don't modify either the program or this documentation."

Since the author is placing a restriction on its distribution (do not modify, keep executable and documentation together) I think his use of the term "Public Domain" is slightly in error. Consider this freely distributable software to be in the 'freeware' vein (copyrighted but no fee requested/required).

Mr. Aldoseri lives in the Persian Gulf country of Bahrain. He can be reached on CompuServe at 75166,2531 for questions, comments, suggestions and updates.

HandyIcons
Freely distributable software by Alan Rubright

There seems to be a raging controversy going on now among Amiga users. It is primarily a battle between CLI and Workbench users. What both sides agree on as a benefit is that, unlike other machines with a graphical user interface, the Amiga gives you a choice. You aren't forced to point and click any more than you are forced to type long command line arguments.

While the CLI proponents wish for more power in that environment the Workbench users are petitioning for more power also. Another common ground of agreement for both groups is that the current Workbench environment is very weak. The arguments centre on just what needs to be done with Workbench to make it much more powerful and intuitive to use. It's fairly widely known that CBM is now working on release 1.4 of the Amiga operating system, and that one major emphasis in 1.4 is a complete rewrite of Workbench. Regardless of what CBM actually produces there will be yells of glee and cries of protest. CBM will not be able to satisfy everyone. So, whether it's now, or after 1.4 is released, the controversy will surely rage on indefinitely.

For the Workbench enthusiast, some very nice programs are available now that can really improve that environment. HandyIcons by Alan Rubright is one of these.

HandyIcons allows you to add a fourth title to your Workbench menu bar. The new entry is appropriately titled 'HandyIcons'.

You can now add your program names, as menu items, to the Workbench menu strip. The present version only supports Workbench Tools, but, hey!, this is a great start. Instead of having to open a disk window, click on a file drawer, and locate the icon for your Tool, you can now add it to the drop down menu of the Workbench title bar.

Under the HandyIcons title on my Workbench you'll find DeluxeMusic, Preferences, Calculator, Clock, and many others. For me, this is so much easier than the alternative the standard Workbench provides. I can install what I want there and not have to be double-clicking on icons, nesting myself deeper and deeper through my subdirectories.

The number of menu entries you can add is limited to the font size you are using and the current vertical resolution of your Workbench. I normally run with an interlaced Workbench using Topaz 11. With that configuration, I can get 47 entries into my HandyIcons menu list.

By the use of an auxiliary program called HandyWB, you can invoke HandyIcons from your startup-sequence. This is for those people like me who enjoy using both the CLI and Workbench environments. I can just boot my machine and have the programs I want quick access to immediately available via the Workbench menus.

This is another of those utilities that I have come to depend upon. I put it in my startup sequence and forget about it. It becomes part of my environment, and many times I forget that it's not really part of the Workbench.

With tools like this, DST, SnipIt, Qmouse, Browser, and a lot of others, you too can customize your system to behave just the way you want and not have to spend the next several months participating in the guessing game of what 1.4 Workbench will be and just how it will be all things to all people.

There is an obvious problem with HandyIcons (at the time of this writing) as distributed by Mr. Rubright. It does not work with the 1.3 ROM installed. (I have not tried it on an A1000 with KickStart 1.3, but I assume the problem exists there also.)

Again, Mr. Khalid Aldoseri comes to the rescue. He has created a patch that will allow HandyIcons to run in machines outfitted with the 1.3 ROM. What the patch does is to simply remove the code in HandyIcons that checks for what version of WB it is running on. With the patch in place, HandyWB works fine with 1.2 and 1.3, though it may run into problems with 1.0 or 1.1.

Here is the patch to fix HandyIcons to work with the 1.3 ROM:

- First, be sure to work on copies, so that if you make a mistake you won't lose the original.
- Using NewZAP (or any disk editor, I use NewZap 3.1), load HandyIcons.
- Search for '$0C000021'

NewZAP will highlight the following values:

0C 00 00 21.

Now, the values after that are:

66 00 01 2a

Change those to:

4E 71 4E 71

and click on SAVE.

- Next, load HandyWB into NewZAP.
- Search for '$0C000021'

Starting from the highlighted values, change the following:

From:

0C 00 00 21 67 1A

To:

0C 00 00 00 66 1A

Again click on SAVE.

Now HandyIcons and HandyWB should work with the 1.3 ROM.

The documentation that is distributed with HandyIcons does not explicitly state if the program is Public Domain, shareware, or freeware. Since no specific statement is made, consider HandyIcons to be freely distributable, copyrighted software by Alan Rubright.

Slideshow Construction Set
Shareware by Adam Rybicki

There are many IFF slideshow programs that are either Public Domain or Shareware. Few, if any, however, are as nice as Adam Rybicki's Slideshow Construction Set (SCS).

SCS allows you to automatically create scripts for displaying your IFF slideshow presentations. SCS is not limited to displaying images of only one type and resolution - each picture is displayed on its own screen. Among the few limitations it *does* have, the current version of the program cannot display overscan or 'extra halfbright' images. Mr. Rybicki plans to add this capability in future versions, however, as well as adding more wipes, fades and other special effects.

You may preview a picture and decide if you want to add it to your script. If you elect to include it, you then choose what effect should be used during the slideshow when the picture is displayed, and what effect should be used after the picture is displayed, before the next one is brought in. The effects you have to choose from are: Fade in, Venetian blind, Checker board, Zoom in, Scroll, Sideways, and Spiral. You may also set the length of time a picture is displayed on a picture by picture basis. The default setting is 3 seconds, and the range of allowable settings is from 0 to 30 seconds. This is the time the program will wait before loading the next picture. Obviously, the loading time differences between floppy disks and hard disks will greatly affect how the delay should be set.

A play program is provided to be used to execute your scripts.

SCS is totally 'Amiga-ized' - it interacts with the user through a very nicely done window full of gadgets. While the concept and overall design of this program is very nice, however, it does have its flaws - for instance, the proportional gadgets in the programs do not work properly. Some areas of the program are not intuitive; that ordinarily wouldn't have been much of a concern, but the supplied documentation is skimpy in many areas.

Overall, this is a nice, usable slideshow program, with just a couple of bugs here and there. I'm looking forward to the next revision, which will I hope will also include more fleshed-out documentation.

Slideshow Construction Set is shareware. No specific fee is requested.

End notes

I mentioned last issue that a couple of very nice CAD programs will soon be available as shareware. This hasn't happened at the time of this writing, but will 'RSN' The programs involved are PCLO, PCLOPlus, and Scheme, all commercial products of SoftCircuits Inc. For various reasons, most of all being very bad luck, SoftCircuits is unfortunately going out of business. The rights to their CAD products belong to Ben Blish, whose contributed a lot of the code to these packages, and whose other programming credits include freeware and shareware programs like Cocomm, FixText, and SetPri.

Mr. Blish is planning to release the CAD products as shareware in the near future (probably by the time you read this), and his new company, Black Belt Systems, will continue to support registered owners. Black Belt Systems is deeply involved with state of the art software for the Amiga and HAM radio operators, including a new (and outstanding) Amiga/HAM slow scan TV software package.

ARexx

The REXX Language for the Amiga

ARexx is a multitasking implementation of the REXX language, an elegant high-level language especially suited for macro-processing. Its clean, simple syntax makes REXX easy to learn for novices — and experienced programmers will appreciate advanced features like compound variables, INTERPRET instruction, and source-level debugging.

- Interactive, Interpreted Operation
- Exceptional String-handling Facilities
- Built-In Library with Over 75 Functions
- Built-In Source-Level Debugger
- Compact, Reentrant code — Only 32K

The Emerging Standard

ARexx is the *de facto* standard for inter-program communication. Software that supports the ARexx command interface can be customized, extended, and combined with other ARexx-compatible software in integrated applications. Watch for announcements of ARexx support from a growing list of Amiga software vendors!

Available Now ... Only $49.95

WShell

The Command Shell You've Always Wanted

WShell is the best command shell on the market, with features and convenience of use that go well beyond the CLI and 1.3 AmigaShell. And yet it's highly compatible with the CLI, so you don't have to learn a new command language to use it!

- Line Editing/Command History/Aliases
- Built-In Commands/Resident Commands
- Prompt String/Window Title "Variables"
- Concurrent Piping
- Transparent Support for ARexx Macros
- Script-bit Support for "Execute" Scripts
- Compact, Reentrant Code — Only 10K

ARexx Command Interface

With WShell you can use REXX-language macro programs like executables — even as filter programs in a piping system. Use the source-level debugging to single-step through a system of concurrently-running "piped" programs!

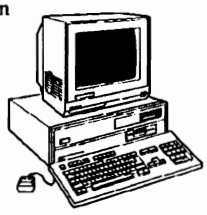
Available Now ... Only $50

Ask your Amiga dealer for ARexx and WShell, or order them directly from the author. Please include a check or money order plus $2/item shipping ($8 for overseas airmail) and any applicable taxes. Dealer inquiries welcome!

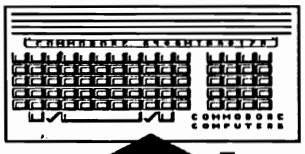
William S. Hawes
P.O. Box 308
Maynard, MA 01754
(508) 568-8695

Amiga is a trademark of Commodore-Amiga, Inc.

GVP Autoboot Hardcards
Mitsubishi Diamond Scan
2058 8 MB cards
Flicker Fixxer
NEC Printers
Super Gen
and more
nothing but the best



Remember..



The Memory Location

Where
Only *AMIGA* makes it possible.

We're Commodore specialists!
Only what works!

COMMODORE AMIGA

Open 10-6 Mon.-Thur. 10-8 Fri.
9-5 Sat. We ship UPS
Commodore authorized repair in store. Low flat rate plus parts.
Warranty service

The Memory Location
396 Washington St
Wellesley, MA 02181
(617) 237 - 6846

RAWCOPY V1.3

- Ultra-powerful backup tool for your AMIGA™
- Copies software that no others will.
- Deprotects many of your favorite programs.
- Supports up to four 3.5" disk drives.
- Mouse-driven user interface.

START	ABORT
SOURCE DRIVE	TARGET DRIVES
DF0: DF1: DF2: DF3:	DF0: DF1: DF2: DF3:

Read Variables		Write Variables	
Drive 0	Read Length=$3000	Drive 1	Write Length=$3000
Cylinder 00	Sync Word=$4489	Cylinder 00	Pre-Write=$0000
Side Bottom Top	Sync On Word ON OFF	Side Bottom Top	Precompensation 00ns 140ns 280ns 560ns
READ TRACK		WRITE TRACK	

Format Control MFM GCR — Bit Cell 2us 4us

See Your Dealer, or write or call for more information. Available on disk for $59.95. Enclose correct amount plus $2.00 for shipping, and specify UPS or first class mail delivery. Visa and MasterCard Charges accepted.

Micro Systems International™
1143 Monroe St.
Carleton, MI 48117
(313) 654-8402

BRiWALL

Our Third Year In Business!

We Know How To Service Our Customers!

SOLID PRODUCTS & SOLID SUPPORT

P.O. Box 129 / 58 Noble Street
Kutztown, PA 19530

24 HOURS — TOLL FREE
1-800-638-5757

OUTSIDE USA CALL 1-215-683-5433

SATISFACTION GUARANTEED

Earn Bonus $$

No Surcharge On Charge Orders

Friendly Service

ACCESSORIES

Control Center	$ 79
Cover 1010 Disk Drive	8
Cover Amiga 1000 CPU	9
Cover Amiga 2000 System	25
Cover Amiga 1000 Keyboard	8
Cover Amiga 2000 CPU	9
Cover Amiga 2000 Keyboard	8
Cover Amiga 500	9
Cover Amiga Monitor	17
Flicker Master	14
Gender Changer/Parallel Adaptor	21
Lightpen for Amiga	89
Modem-Mini A1000	79
Modem-Mini A500/A2000	79
Modem Supra w/o cable	139
Modem Cable-RS232 to A1000	22
Modem Cable for A500	22
Okimate Plug N Print-Amiga	75
Ribbons	Call

BOOKS

1001 Things to do with Amiga	11
Advanced Amiga BASIC	15
Amiga BASIC Inside & Out†	20
Amiga DOS Express	20
Amiga DOS Inside & Out†	17
Amiga DOS Manual	20
Amiga DOS Reference Manual	19
Amiga Disk Drives Inside & Out	24
Amiga Hardware Manual	18
Amiga Machine Language†	17
Amiga Programmer's Guide	15
Amiga System Programmer's Guide	27
Amiga Tips & Tricks†	17
Amiga for Beginners	15
Beginner's Guide/Amiga	15
C Primer Plus	21
Elementary Amiga BASIC	13
Inside Amiga Graphics	15
Inside the Amiga With C	21
Intuition Manual	20
Kickstart Guide to Amiga	19
Machine Language-Compute	17
ROM Kernal : Exec	18
ROM Kernal : Lib/Dev	24
Abacus Disks each	12

GENERAL BUSINESS

Acquisition	189
Analyze 2.0	69
Becker Text	99
Critic's Choice	169
Data Retrieve	52
Data Retrieve Professional	199
Excellence	169
Family Tree (Your)	35
Flow	65
Goldspell 2	32
Haicalc	39
KFS Accountant	195
Keepin Track	45
Kindwords	69
Logistix	69
Magellian	139
MaxiPlan Plus	125
MaxiPlan 500	95
Microfiche Filer	69
Microfiche Filer Plus	125
Microlawyer	39
Money Mentor 2.0	69
Organize	49
P.H.A.S.A.R.	69
Page Setter	90
Professional Pages 1.1	249
Proscript for Prowrite	32
Prowrite v 2.0	75
Publisher Plus	64
Publishing Partner	$129
Reason	249
Scribble	55
Security Analyst	65
Shakespeare	149
Softwood File II SG	69
Softwood Write & File	69
Superbase Personal	59
Superbase Personal 2	105
Superbase Professional	235
TX ED Plus	59
Textpro	59
Viziwrite	59
Word Perfect	199
Word Perfect Library	99
Works (The)	125
Works - Platinum Edition*	175

EDUCATIONAL

Adventures of Sinbad	32
Aesop's Fables	32
All About America	39
Decimal Dungeon	32
Discovery Math	27
Discovery Spell	27
First Letters/Words	30
First Shapes	30
Fraction Action	32
Ghostly Grammer	32
Kid Talk	30
Kinderama	32
Linkword: French	24
Linkword: German	24
Linkword: Italian	24
Linkword: Spanish	24
Logic Master	32
Math Talk	30
Math Talk FRACTIONS	30
Math Wizard	32
Math-amation (College)	55
Mavis Beacon Typing	36
Perfect Score SAT	55
Planetarium*	49
Read A Rama	32
Speller Bee	29
Surgeon	34
Word Master	32

ENTERTAINMENT

4x4 Off Road Racing	29
AAArgh	25
Annals of Rome	27
Arkanoid	20
Auto Duel	35
Awesome Arcade Pak	36
Balance of Power	36
Barbarian	28
Bard's II: Destiny Knight	39
Bard's Tale I	33
Bard's II: Clue Book	12
Battle Chess	39
Better Dead then Alien	22
Beyond Zork	34
Black Cauldron	29
Blackjack Academy	27
Breach	27
Bridge 5.0	25
Bubble Ghost	27
Capone	$ 27
Carrier Command	32
Centerfold Squares	24
Chessmaster 2000	36
Clever & Smart	22
Corruption	32
Craps Academy	32
Defender of the Crown	33
Descartes	25
Double Dragon*	29
Dragon's Lair*	35
Dungeon Master	28
Earl Weaver Baseball	35
Earl Weaver's Commissioner's Disk	17
Ebonstar	27
Empire	36
F18 Interceptor	36
Faery Tale Clue Book	9
Faery Tales	32
Ferrari Formula One	36
Fire & Forget	29
Fire Power	19
Fire Zone	27
Flight Simulator 2	35
Fourth & Inches	29
GB Air Rally	25
Galactic Invasion	19
Galileo 2.0*	49
Gone Fish'n*	35
Guild of Thieves	29
Gunshoot	22
Hardball Baseball	29
Hunt for Red October	36
Hybris	27
Ice Hockey-Super Star	35
Ice Hockey: Wayne Gretsky	35
Jet	35
King of Chicago	33
Kings Quest III	36
Knight Orc	29
Leaderboard World Class	29
Leisure Suit Larry	29
Lord of the Rising Sun	33
Main Frame	28
Marble Madness	36
Master Ninja*	29
Mean 18 Golf	27
Mobieus	39
Obliterator	28
POW	29
Phasar Gun	42
Port of Calls	32
Portal	27
Pub Games*	27
Questron 2	36
Reach for the Stars*	29
Replay 4	29
Rocket Ranger	33
Roger Rabbit	33
Romantic Encounters - Dome	27
Rubicon Alliance*	24
SDI	33
Sargon 3	29
Scrabble	29
Sex Vixens Hint Sheets	5
Sex Vixens from Space	29
Shadow Gate	33
Shangai	27
Sherlock Holmes: Crown Jewels	29
Sky Fox 2	29
Space Quest 2	36
Space Station Oblivion	36
Spacecutter	$ 25
Star Fleet 1	39
Star Glider 2	32
Stellar Conflict	27
Strip Poker	29
Sword of Sodan	34
TV Sports Football*	33
Terror Pods	28
Test Drive	27
Thexder	25
Three Stooges	33
Tracers	27
Turbo	19
Twilight's Ransome*	27
Ultima 3	33
Ultima IV	39
Universal Military Simulator*	35
Virus	25
Wizard Wars	29
World Tour Golf	29
Zany Golf*	33
Zoom	20

CREATIVITY/GRAPHICS

Analytic Art	40
Animate 3D	99
Animation Effects	39
Animation Flipper	45
Animation Stand	39
Animator	85
Bumper Sticker Maker	37
Business Card Maker	37
Calligrapher v 1.05	79
Celebrity Cookbook	27
Comic Set-Funny Figures Clip Art	25
Comic Set-Science Fiction Clip Art	25
Comic Set-Super Heroes Clip Art	25
Comic Setter	69
Deluxe Paint 2	86
Deluxe Photo Lab	99
Deluxe Print 2	55
Deluxe Productions	139
Deluxe Video 1.2	86
Design Disks each	25
Digipaint 3.0	40
Digiview 3.0	129
Digiview Gold for A500/A2000	129
Director	49
Draw 2000	159
Express Paint V3	69
Fantavision	49
Flipside	25
Graphics Studio	39
Home Builder's CAD	135
Images	29
Intro CAD	52
Invision	99
Lights, Camera, Action	59
Modeler 3-D	69
Page Flipper	39
Page Flipper Plus FX	99
Page Render 3D	99
Page Sync	69
Photon Paint	65
Photon Paint Surface Disk	25
Photon Video Cel Animator	99
Photon Video Transport Controller	195
PrintMaster Fantasty Art Gallery	22
PrintMaster Art Gallery 1 or 2	22
PrintMaster Fonts & Borders	25
Printmaster Plus	$ 33
Pro Video Plus	199
Sculpt 3D	69
TV Show	65
TV Text	65
Turbo Silver Plus	139
Ultra CAD	169
Video Scape 3D 2.0	125
Video Titler 1.1	99
X-Specs	92
Zoetrope	89
Zuma Fonts-each	25

SOUND/MUSIC

Deluxe Music Construction	65
Dr T Midi Recording Studio	49
Dr. T KCS V1.6A	165
Dynamic Drums	52
Dynamic Studio	135
Instant Music	36
Midi Magic	99
Model A Midi Interface	49
Music-x	195
Perfect Sound A1000	65
Perfect Sound A500	65
Sonix	52
Sound Oasis	65
Sound Sampler	85
Soundscape Pro Midi Studio	129
Studio Magic	65
Synthia	65

UTILITIES/LANGUAGES

64 Emulator 2/Amiga 500	52
64 Emulator for Amiga 1000	52
A Talk Plus	52
AC BASIC v 1.3	135
AC FORTRAN	199
Amiga Workbench 1.3	24
Assempro	64
Aztec C Developer Compiler	195
Aztec C Professional Compiler	129
Aztec C Source Level Debugger	49
BBS PC/New Version	99
Benchmark Modula 2	129
Benchmark Modula 2-Simp A Lang	69
Benchmark Modula 2-IFF Library	69
Butcher 2.0	25
C.A.P.E. 64K	59
CLI Mate	27
Cygnus Ed Prof Text Editor	69
DOS-2-DOS	39
DSM	49
Diga	52
Disk-2-Disk	35
Diskmaster	39
FACC II	25
GOMF V3.0	27
GOMF V3 w/Hardware	49
Grabbit	20
Inova Tools	52
Lattice C + +	339
Lattice C MS-DOS/AmigaDOS Comp.	359
Lattice C V 5.0 Developer Pack	205
Mac Emulator*	call
On Line V2	42
Power Windows 2.5	59
Project D	35
Quarterback	45
True Basic Libraries each	35
True Basic 2.0	69
True Basic Runtime Pak	99
UltraDOS	39
Virus Infection Protection	33

IT IS OUR POLICY to stock what we advertise and carry the best products available at the best prices possible. OVER 500 PRODUCTS in stock now and more arriving daily!! IF YOU DON'T SEE IT LISTED, PLEASE GIVE US A CALL.

Products marked with * were not released at time ad was prepared. Please call for firm price and availability.

HARDWARE

CALL US!

Let us put a custom package deal together for you!

Please Call M-F, 9-4 EST

AmiGEN Genlock	$149
Bill's Boards Bus Expander	484
CMI VI-500 video adapter for A500	68
Color Splitter	75
Copy Stand	59

DRIVES

AIR 3.5" External	189
AIR 3.5" Internal	145
Mast 3.5" Internal	129
Mast 3.5" Twin External	259
Mast 3.5" Unidrive	149
Phoenix 22 Meg/A1000	579
Phoenix 22 Meg/A500	579
Phoenix 48 Meg/A1000	829
Phoenix 48 Meg/A500	829
Supra 22 Meg/A1000	659
Supra 22 Meg/A500	659
Supra 30 Meg/A1000	729
Supra 30 Meg/A500	729

Actual Shipping Costs on Hardware

ECE Midi 1000	49
ECE Midi 500/2000	49
Easyl 1000 Draw Tablet	369
Easyl 2000 Draw Tabler	369
Easyl 500 Draw Tablet	329
Flicker Fixer/A2000	489
Frame Grabber	549

GREAT VALLEY

A2000 20MB Hard Card SCSI	619
A2000 2MB Hard Disk Cntlr SCSI	309
A2000 30MB Hard Disk Card SCSI	679
A2000 40MB Hard Card Quantam	819
A2000 45MB hard Card SCSI	769
A2000 80MB Hard Card Quantam	1159
A2000-2/0 w/80MB Quantam Drv	1249
A500 20MB Hrd Drv/RAM Subsyst	699
A500 30MB Hrd Drv/RAM Subsyst	759
A500 40MB Q Hrd Drv/RAM Sbs	1020

Gen/One Genlock	719
IVS Trumpcard SCSI Controller	159
Live! A1000 Framegrabber	249
Live! A2000 Framegrabber	335
Live! A500 Framegrabber	285

MEMORY BOARDS

minimegs A1000 (OK)	165
Minimegs A500 (OK)	165
Pro RAM/A2000/OK	249
RC4 RAMcard/A1000/OK	199
RC4 RAMcard/A500/OK	199
Ronin M1000	519
Ronin M2000	519
Spirit-A1000 (OK)	240
Starboard II (OK)	269
Starboard II (1 Meg)	659
Starboard II (512K)	459

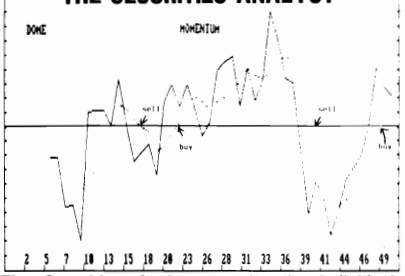
Panasonic 16mm Lens NO iris	22
Panasonic 16mm Lens with Iris	49
Panasonic W1410 Camera	189
Perfect Vision	169

PRINTERS

Canon PJ1080A Clr Ink Jet	519
Hewlett Packard Desk Jet	779
Hewlett Packard Paint Jet	1150
NEC LC890 Poscript Laser	3499
Okimate 20	155
Okimate Plug & Print	55
Panasonic 1019i	239
Panasonic 1092	379
Star NX-1000R Rainbow	269
Toshiba Page Laser 12	3159
Xerox 4020 Color Ink Jet	1199

ProGEN Gen Lock	379
Ronin 68020-12 Microprocessor	189
Ronin 68030 adapter board	279
Ronin 68030 microprocessor	549
Ronin 68881-16 Math co-processor	189
Ronin 68881-20 Math co-processor	285
Ronin 68881-25 Math co-processor	579
Ronin H1000 Accelerator Board	345
Ronin Hurricane H2000	519
Subsystem 500	199
Supergen	689
Tiny Tiger Hard Drive Card	229

THE SECURITIES ANALYST



The Securities Analyst permits the individual investor to chart any number of stocks over an extended period of time. Using this package you can make intelligent buy-sell decision based on analysis of market trends. Analysis charts may be displayed on the screen or output to printer. A comprehensive manual on Technical Analysis of Stock Market Trends is also included. Great Buy!

INTRO OFFER $65

BONUS DOLLARS

EARN BONUS $$$ WHEN YOU ORDER!!! For every $50 of software and accessories that you order, you earn one bonus dollar. That's an additional 2% discount!! Use your bonus dollars on future purchases.

CATALOGS

We now have our ALL NEW AMIGA catalog, describing hundreds of products for your computer. Call or write for your copy of our AMIGA CATALOG today!!

BRiWALL

SOLID PRODUCTS & SOLID SUPPORT
P.O. Box 129 / 58 Noble Street
Kutztown, PA 19530
24 HOURS — TOLL FREE
1-800-638-5757

Circle 132 on Reader Service card.

As Captain Brad Stallion, you have been assigned a high priority mission by the Federated Government which has long been harried by a series of raids on the male population of its colonies. It is rumored that these raids are conducted by "The Tribe", a colony of beautiful female clones from the mysterious planet Mundo. Your mission is to locate the colony and destroy it! Please be advised that this is an ADULT adventure game!!

(Hint Sheets - $5)

ONLY $29

A Superior Word Processor!

Specifically designed for your AMIGA!

Experience

excellence!

No other product has all the powerful features required of a modern word processor, in a package sophisticated enough to use in desktop publishing! WYSIWYG, Spellcheck (as you type), Thesaurus, Multiple columns, Color, Graphics, Fonts, Math, Postscript Output and MORE!

SPECIAL PRICE! $139

(Good until 2/28/89)

SIZZLING HOT ENTERTAINMENT TITLES

Battle Chess	$39
Chrono Quest	35
Dragon's Lair	35
Dungeon Master	28
Heroes of the Lance (if released)	29
Pioneer Plaque	29
Pub Games	27
Rocket Ranger	33
Sword of Sudan	34
TV Sports Football	33
Ultima IV	39
Universal Military Simulator	35
Wayne Gretsky Hockey	35
Who Framed Roger Rabbit	33

BUY ANY 3 of the above titles and take an ADDITIONAL $5.00 off of your order!

(Offer good until 2/28/88)

ATTENTION! NEW AMIGA OWNERS

We want your business!

We will give you $10 off on your first software order if:

1.) Your order is $100 or more.

2.) You prepay with a check or money order.

3.)You send us a sales receipt for an A500, A1000 or A2000 dated between 12/1/88 and 1/31/88.

Offer good until 2/28/88.

Not valid with any other specials.

Come visit us! M-F, 9-5 EST, Sat. 9-12

New FAX (215) 683-8567

Prices subject to change without notice. Normally they don't, but this makes it legal. If in doubt, call!

ORDERING INSTRUCTIONS

For your convenience, you can place a phone order 24 HOURS/7 DAYS A WEEK on our TOLL-FREE LINE 1-800-638-5757. Price, Availability and Compatability Checks are also Welcome on our order line, Monday thru Friday, 9AM-4PM EST. AFTER HOURS, Orders Only Please!! When placing an order, please specify your COMPUTER MODEL, HOME & DAYTIME PHONE NUMBER, SHIPPING ADDRESS, METHOD OF PAYMENT and ITEMS ORDERED. To help us serve you better, please have all your information, including your CHARGE CARD number, ready before you call us.

ACCEPTED PAYMENT METHODS: We gladly accept payment by, PREPAID BY PERSONAL CHECK (will not hold for clearing) or MONEY ORDER; COD (continental USA only AND $4.00 ADDITIONAL); MASTERCARD; VISA; SCHOOL PURCHASE ORDERS. All payment must be in USA DOLLARS. THERE IS NO SURCHARGE FOR CHARGE CARDS and your card is NOT charged until we ship.

TECHNICAL SUPPORT

We do our very best to help you with your product selections, before you order and after you receive your product. General questions, price, and compatibility with your computer, etc. will be handled by our order staff at the time you place your order. BUT if you have specific, detailed questions about a product, printer compatibility questions etc., you will get the most help from our TECHNICAL SUPPORT LINE at 215-683-5999. Call Monday thru Friday, 9AM-4PM EST, and our trained tech staff will be happy to help you.

SHIPPING POLICY

ALL ORDERS received before 3PM EST will normally be shipped same or next business day. Out-of Stock items will be shipped by the same shipping method as original order, normally within 3 or 4 business days. All UPS shipments are sent SIGNATURE REQUIRED/NO DRIVER RELEASE. ADD the following shipping charges to your TOTAL software order.

UPS: GROUND-$4.00 (cont USA ONLY); AIR/RUSH-$7.00 (Includes Puerto Rico/Hawaii); OVERNIGHT-$12.00 PLUS $2.50 per item (must be received by 12 NOON)

PRIORITY MAIL: USA-$4.00 (includes APO/FPO); CANADA/MEXICO-$6.00; OTHER FOREIGN-$4.00 handling PLUS Actual Shipping (minimum $12.00)

RETURN POLICY

We have a liberal return policy to better service your needs. Software piracy is a problem, but as long as our policy is not abused, we will continue to honor it. If within 15 days from the time you receive an item, you are not satisfied with it for any reason, you may return it to us for either a refund, exchange or open credit. REFUNDS ARE SUBJECT TO A10%RESTOCKING FEE PER ITEM RETURNED ($5.00 MINIMUM PER ITEM). A DEFECTIVE ITEM will be replaced with the same item (NO CHARGE... of course). EXCHANGES/OPEN CREDITS will gladly be issued for the FULL PURCHASE PRICE OF THE ITEM.

Solid Modelling

Algorithmic generation of Sculpt 3D scenes

by Cathryn and Eric Graham

Eric and Cathryn have been involved with three-dimensional modelling on the Amiga for over two years, producing the Juggler Demo, Sculpt 3D and Sculpt Animate, as well as magazine articles, and the Ray Tracing Newsletters. They live in a high mountain valley in Colorado that they share with elk, coyote, an occasional eagle and other reclusive denizens. They can be reached at P.O. Box 8298, Durango, Colorado 81301. A disk containing the source code in this article, and other good stuff is available for $6.00, including postage.

Thanks to the many solid modelling, ray tracing and animation packages available for the Amiga, it has become the premier computer for generating and using three dimensional shapes. In this article we will show you how to write programs that generate the data to describe many different solid shapes.

The solid modelling programs allow you to draw freehand curves in two dimensions and then spin them to produce surfaces of revolution, or extrusions. Some will even produce spiral or helical surfaces, or allow you to join cross-sectional slices into a single object. Simple geometrical shapes like cubes, cylinders and spheres are usually provided as primitive objects. Shapes of any complexity can be hand-built from individual triangles, but this is time consuming and requires great skill.

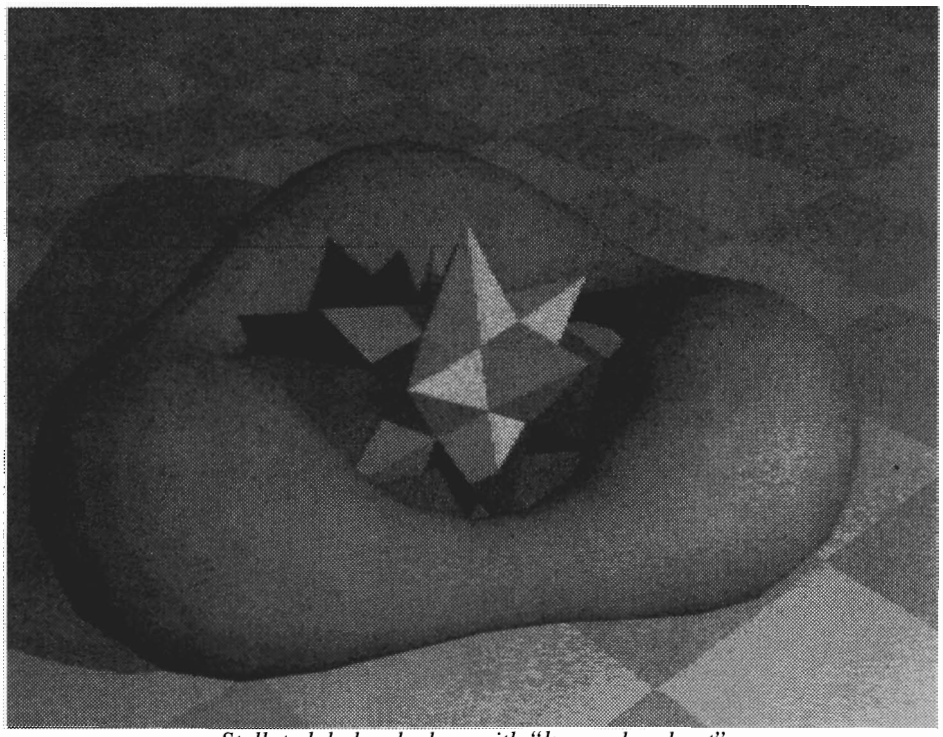
There are still many shapes that are regular enough to be described mathematically, but which cannot easily be produced with available solid modelling programs. This is where it sometimes pays to write a custom program to produce the shape.

Representing solid objects

There are many ways to represent a solid object in a computer, but we will choose the one used by the Sculpt 3D family of programs, both because it is particularly simple, and because programs exist to convert its data format to those of most other rendering programs.

The surface of a solid object can be represented, either exactly or approximately, by a net of triangular elements. Each element is called a face, the sides of faces are called edges and the corners are called vertices (the plural of vertex).

The position of each vertex can be specified by three numbers, for example the distances east, north and above some reference position. The locations of the vertices provide all the geometric information to describe a shape, but some additional linking data is required to specify which vertices are to be joined by edges. For simplicity, let us assume that a face exists wherever three edges form a triangle. The faces can carry additional information, such as the colour or texture of the surface. There are three stages in generating data for a solid body. First you must decide how many vertices are needed and cal-



Stellated dodecahedron with "lumpy doughnut"

culate their positions. Call this the geometrical stage. Then you must decide how the net of edges are to be connected. This is called the topological stage. Finally the data must be prepared in an output format that is understandable by other programs. Fortunately, this last stage is the easiest.

Topology

Topology is a branch of mathematics that is concerned with the properties of objects that are independent of measurement. No matter how you stretch the inner tube of a tire, you will never be able to turn it into a beachball unless you use scissors and glue. There is something essentially different between the surface of a sphere and that of a torus. So it is also, when surfaces are represented by a triangular grid. Unless you delete edges and reconnect vertices, some shapes cannot be converted to others simply by moving the vertices.

One interesting property of a grid of triangles is called its Euler number, which is simply the sum of the number of faces and vertices, less the number of edges. Any grid drawn on a surface that is topologically equivalent to a sphere has an Euler number of 2. 'Topologically equivalent' is a mathematical term meaning that one shape can be turned into another by stretching or bending. Any grid that is topologically equivalent to a finite open surface has an Euler number of 1. The Euler number for a torus is 0. You can use Euler numbers as a check for errors when you are building complicated objects one vertex at a time with a solid modelling program.

Data structures for solid modelling

In order to represent vertices, edges and faces in a computer you must maintain data such as:

Which vertices lie at the ends of an edge.
Which vertices serve as the corners of a face.
Which edges connect to a vertex.

The last item presents a difficulty, because there is no limit, in principle, to how may edges meet at a vertex. If we used a fixed size array to store this information, we would either be restrictive or wasteful, unless you knew ahead of time what size to make the array. This suggests that we avoid the use of arrays and design a data structure using linked lists, where one item is linked to another by means of a pointer. In a language like C, this is particularly easy. The following C structures can be used for solid modelling:

```
struct vertex
   {struct vertex *next;
    double pos[3];      /* Coordinates of vertex */
    struct edgelist *edgelist;
    long int index;
   };
struct edgelist
   {struct edgelist *next;
    struct edge *edge;
   };
struct edge
   {struct edge *next;
    struct vertex *vertices[2];
   };

struct face
   {struct face *next;
    struct vertex *vertices[3];
    long int properties;
   };
```

Each structure has a spare field called 'next' which is used to point to the next structure of the same kind. This ensures that we never lose an item, unless we forget which is the first one on the list. The problem of the number of edges meeting at a vertex is solved by the structure edgelist. It is an element in a list that contains a pointer to an edge, so any number of edges can be associated with each vertex.

Notice that the vertex structure contains the field 'pos' that defines the position of the vertex. All the geometric data for an object resides in these fields. The face structure contains a field called 'properties'. In Sculpt 3D, the first 3 bytes are used to store the red, green and blue components of the colour of the face, while the last byte contains information about the surface texture. Within the last byte, the most significant bit is set to imply that the technique of Phong shading is to be used to make the surface seem smooth, when rendered. The remaining bits represent additional surface information, for example the value 0 represents a dull surface and 1 represents a shiny surface.

Whenever you need a new structure, you can use the following functions to allocate memory and initialize the structures.

```
struct vertex *newvertex(x, y, z, next)
double x,y,z;
struct vertex *next;
{struct vertex *v;
 char *malloc();

   v=(struct vertex *)malloc(sizeof(struct vertex));
   v->pos[0]=x; v->pos[1]=y; v->pos[2]=z;
   v->edgelist=0; v->next=next;
   return v;
}

struct edgelist *newedgelist(edge, next)
struct edge *edge;
struct edgelist *next;
{struct edgelist *el;
 char *malloc();

   el=(struct edgelist *)malloc(sizeof(struct edgelist));
   el->edge=edge;
   el->next=next;
   return el;
}
```

```
struct edge *newedge(v1, v2, next)
struct vertex *v1,*v2;
struct edge *next;
{struct edge *e;
 char *malloc();

   e=(struct edge *)malloc(sizeof(struct edge));
   e->vertices[0]=v1; e->vertices[1]=v2;
   e->next=next;
   return e;
}

struct face *newface(v1, v2, v3, properties, next)
struct vertex *v1,*v2,*v3;
long int properties;
struct face *next;
{struct face *f;
 char *malloc();

  f=(struct face *)malloc(sizeof(struct face));
  f->vertices[0]=v1; f->vertices[1]=v2; f->vertices[2]=v3;
  f->properties=properties;
  f->next=next;
}
```

Each function has an argument that lets you set up the 'next' field. In the interest of brevity, the value returned by the memory allocation function is not tested. If no memory is available, a zero result is returned. If the code is not modified to make this test, you can be assured of a visit from the Guru if you run out of memory!

Finally we need a structure to tie together all the lists that describe a single object. Appropriately we call it 'object':

```
  struct object
     {struct vertex *vertices;
      struct edge *edges;
      struct face *faces;
     };
```

Building an object

Now we can look at how we might build a simple object like a cube. The first stage is to create the vertices and set them at the correct locations. Then the edges must be specified, remembering to place a diagonal edge across each square face, since Sculpt 3-D only recognizes triangular polyhedra. Here is how it could be done

```
makecube(object, properties)
struct object *object;
long int properties;
{int i,j,k;
 struct vertex *v[2][2][2], *newvertex();

object->vertices=0; object->edges=0; object->faces=0;
for (i=0; i<2; ++i)
  for (j=0; j<2; ++j)
    for (k=0; k<2; ++k)
      object->vertices=v[i][j][k]=
        newvertex((double)i, (double)j, (double)k, object->vertices);

for (i=0; i<2; ++i)
  for (j=0; j<2; ++j)
    for (k=0; k<2; ++k)
       {if (!i) link(object,v[0][j][k],v[1][j][k],properties);
        if (!j) link(object,v[i][0][k],v[i][1][k],properties);
        if (!k) link(object,v[i][j][0],v[i][j][1],properties);
        if ((!j) && (!k)) link(object,v[i][0][0],v[i][1][1],properties);
        if ((!i) && (!k)) link(object,v[0][j][0],v[1][j][1],properties);
        if ((!i) && (!j)) link(object,v[0][0][k],v[1][1][k],properties);
       }
}
```

This is a particularly unpleasant function for such a simple object, particularly since a cube can be generated as a primitive object by most solid modelling programs with a single mouse stroke. Take heart - very much more complicated and interesting objects can be built with hardly any more effort.

Note that we have hidden the details of edge and face construction in the function 'link'. This function creates an edge between two vertices, and links it into the object structure and makes the edge known to the two vertices. It is also on the alert for edges that might cause a triangle to be completed. If so, a face is created and given the appropriate surface properties. Here is how it works:

```
link(object, v1, v2, properties)
struct object *object;
struct vertex *v1,*v2;
long int properties;
{struct edge *e,*newedge();
 struct edgelist *el,*el2,*newedgelist();
 struct vertex *v3,*otherend();
 struct face *newface();

/*  does edge already exist? */
for (el=v1->edgelist; el; el=el->next)
   if (otherend(el->edge,v1) == v2) return;      /* yes! */

e=object->edges=newedge(v1,v2,object->edges);   /* the edge is made */
v1->edgelist=newedgelist(e,v1->edgelist);       /* link to first vertex */
v2->edgelist=newedgelist(e,v2->edgelist);       /* and to second */

/* does this edge complete any faces? */
for (el=v1->edgelist; el; el=el->next)
   {v3=otherend(el->edge,v1);
    if (v3 == v2) continue;  /* skip new edge */
    for (el2=v3->edgelist; el2; el2=el2->next)
       if (otherend(el2->edge,v3) == v2)         /* make a face? */
          object->faces=newface(v1,v2,v3,properties,object->faces);
   }
}
```

The rather strangely named function 'otherend' does what its name suggests. Given an edge and a vertex, it returns a point to the vertex at the other end of the edge from the given vertex. The following code will suffice:

```
struct vertex *otherend(e,v)
struct edge *e;
struct vertex *v;
{if (e->vertices[0] != v) return e->vertices[0];
 return e->vertices[1];
}
```

Output data

The data representing a cube would be of no use if it could not be changed into a form suitable for other programs. All our effort would be of no avail. Objects in Sculpt 3D are described by a "scene" file. All scene files have names that end in '.scene'. The data is stored in the form of an Interchange File Format (IFF). The mechanics of reading and writing IFF files are described in the ROM Kernel Manual as well as in documents that can be found in most Amiga public domain disk collections. It is generally much easier to write an IFF file than to read one, because we get to choose what data is included. IFF files are built up from 'chunks' of data. A chunk has three parts: a four byte identifier, a four byte count of the number of bytes that follow in the chunk, and the actual data.

An IFF scene file consists of a single chunk called "FORM". The data of this chunk consists of the name "SC3D", which identifies the file as a Sculpt 3D scene file and one or more other chunks. There are many possible chunks that can be found in a scene file, for example chunks containing hierarchy data, cubic spline and motion paths, but we are only concerned with three of them: "VERT", "EDGE" and "FACE". As you could guess, they contain the data for vertices, edges and faces. The VERT chunk contains one record for each vertex. The record consists of three 32 bit integers representing the coordinates of the vertex. Because a 32 bit integer can represent a number as large as a billion, it is a good idea to scale your coordinates so that they become large integers - values around 100,000 work fine.

The EDGE chunk also has one record for each edge, but the layout for the record is different, consisting of two 32 bit integers that represent indices for the vertices at each end of the edge. An index value of 0 refers to the first vertex in the VERT chunk, a value of 1 refers to the second vertex, etc.

The FACE chunk is also comprised of records. Each record starts with three 32 bit indices that denote the vertices at the corner of the face. The last item in the record is a 32 bit number representing colour and surface properties, in the format described earlier.

The following function will produce an IFF scene file corresponding to an object:

```
wrscene(name, object)
char *name;
struct object *object;
{long int l1,l2,l3,l4,ne,nv,nf,pos[3];
 int j;
 struct vertex *v;
 struct edge *e;
 struct face *f;
 FILE *fopen(),*file;

 /* Find the lengths of the lists */
 for (nv=0,v=object->vertices; v; ++nv,v=v->next)
    v->index=nv;
 for (ne=0,e=object->edges; e; ++ne, e=e->next);
 for (nf=0,f=object->faces; f; ++nf, f=f->next);
 l2=12 * nv;
 l3= 8 * ne;
 l4=16 * nf;
 l1=12 + l3 + l4 + 28;

 file=fopen(name,"w");     /* You should check that the file is OK! */
 fwrite("FORM",4,1,file);
 fwrite(&l1,4,1,file);
 fwrite("SC3DVERT",4,2,file);
 fwrite(&l2,4,1,file);
 for (v=object->vertices; v; v=v->next) /* VERT chunk data */
     {for (j=0; j<3; ++j)
      pos[j]=100000.0*v->pos[j];
      fwrite(pos,4,3,file);
     }

 fwrite("EDGE",4,1,file);
 fwrite(&l3,4,1,file);
 for (e=object->edges; e; e=e->next)   /* EDGE chunk data */
    for (j=0; j<2; ++j)
       fwrite(&(e->vertices[j]->index), 4, 1, file);

 fwrite("FACE",4,1,file);
 fwrite(&l4,4,1,file);
 for (f=object->faces; f; f=f->next)   /* FACE chunk data */
    {for (j=0; j<3; ++j)
     fwrite(&(f->vertices[j]->index), 4, 1, file);
     fwrite(&(f->properties),4,1,file);
    }
 fclose(file);
}
```

To use the functions described above to create a cube, you need a main function as follows:

```
#include "stdio.h"

main()
{struct object object;

 makecube(&object, 0xFFFFFF00);
 wrscene("cube.scene", &object);
}
```

The lumpy doughnut

Before you give up in despair at the amount of work required to perform an essentially trivial operation, let us show you how only a little extra work is needed to make something much more interesting: a lumpy doughnut.

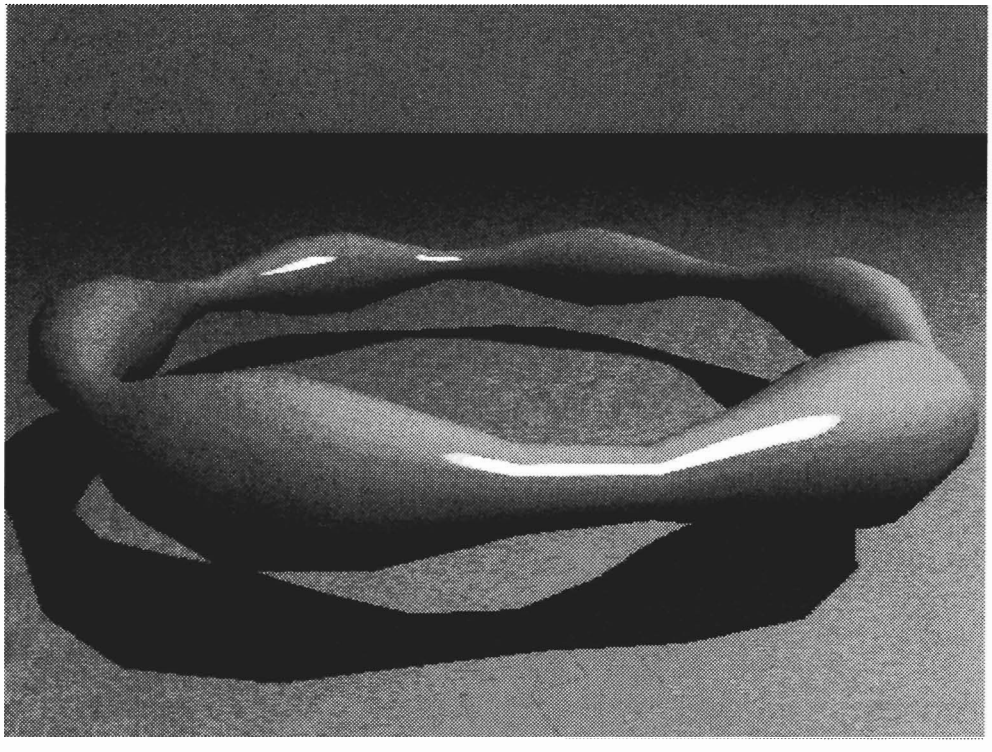
Before adding lumps, take a moment to think about how you might inscribe little triangles all over a regular doughnut. One way would be to draw a series of n1 rings, equally spaced along the length of the doughnut. Each ring is made up of a loop of n2 edges and n2 vertices. Finally, join the vertices on adjacent rings to form a net of triangles. Here is how it is done:

```
makedonut(object, properties)
struct object *object;
long int properties;
{double r1,r2,pi,da1,da2,a1,a2,xc,yc,dx,dy,dz;
 int i,j,n1,n2;
 double sin(),cos();
 struct vertex *ringfirst,*ringlast;

 object->vertices=0; object->edges=0; object->faces=0;

 r1=1.0; r2=0.3;
 n1=24;  n2=10;                /* for example */
 pi=3.1415927;
 da1=2.0*pi/n1; da2=2.0*pi/n2;

 for (a1=0.0,i=0; i<n1; a1=a1+da1,++i)        /* big circle */
    {xc=r1*cos(a1); yc=r1*sin(a1);
     for (a2=0.0,j=0; j<n2; a2=a2+da2,++j)     /* small circle */
        {dx=r2*cos(a2); dz=r2*sin(a2);
         dy=dx*sin(a1); dx=dx*cos(a1);
         dx=dx+xc; dy=dy+yc;
         object->vertices = newvertex(dx,dy,dz,object->vertices);
        }
     linkring(object, object->vertices, n2, properties);
     if (i)
        linkrings(object, object->vertices, ringlast, n2, properties);
     else ringfirst=object->vertices;
     ringlast=object->vertices;
    }
 linkrings(object,ringlast,ringfirst,n2,properties);
}
```

A little trigonometry is needed to calculate the circles. The big circle passes through the centers of the rings that we will create - this is calculated by the first pair of sin and cos functions. The second sin cos pair calculates the small circle that represents the ring. The final pair of sin cos functions rotates the small circle to the correct orientation. So much for the geometry, now for the topology.

Because the vertices are linked together by their 'next' fields as they are created, we can refer to a ring by its youngest vertex, provided we know how many vertices are on the ring, in our case that is n2. The function 'linkring' joins the vertices into a ring, remembering to link the first and last vertices. Here is the 'linkring' function:

```
linkring(object,vring,n,properties)
struct object *object;
struct vertex *vring;
int n;
long int properties;
{struct vertex *v,*previous;
 int i;
```



Doughnut with six lumps

```
 for (v=vring, i=0; i<n; ++i)
    {if (i) link(object, v, previous, properties);
     previous = v;
     v = v->next;
    }
 link(object, vring, previous, properties);
}
```

To link together adjacent rings, we first link the vertices that are closest neighbors and then fill in with some extra edges to split each rectangle into a pair of triangles:

```
linkrings(object, ring1, ring2, n, properties)
struct object *object;
struct vertex *ring1, *ring2;
int n;
long int properties;
{struct vertex *v1,*v2,*previous2;
 int i;

 for (v1=ring1, v2=ring2, i=0; i<n; ++i)
    {link(object, v1, v2, properties);      /* nearest */
     if (i) link(object, v1, previous2, properties);  /* triangulate */
     previous2 = v2;
     v1 = v1->next; v2 = v2->next;
    }
 link(object,ring1,previous2,properties);  /* loose end! */
}
```

The main function should be modified to call the function to make and save the doughnut:

```
main()
{struct object object;
 makedonut(&object, &0xF0C00081); /* smooth and shiny */
 wrscene("donut.scene", &object);
}
```

Of course this is only a regular doughnut, and not the lumpy one that was promised. To make it lumpy, all you have to do is to modulate r2, the radius of the small circle, as you go around the big circle. Declare rl to be a double, then replace the line:

dx = r2 * cos(a2); dz = r2 * sin(a2);

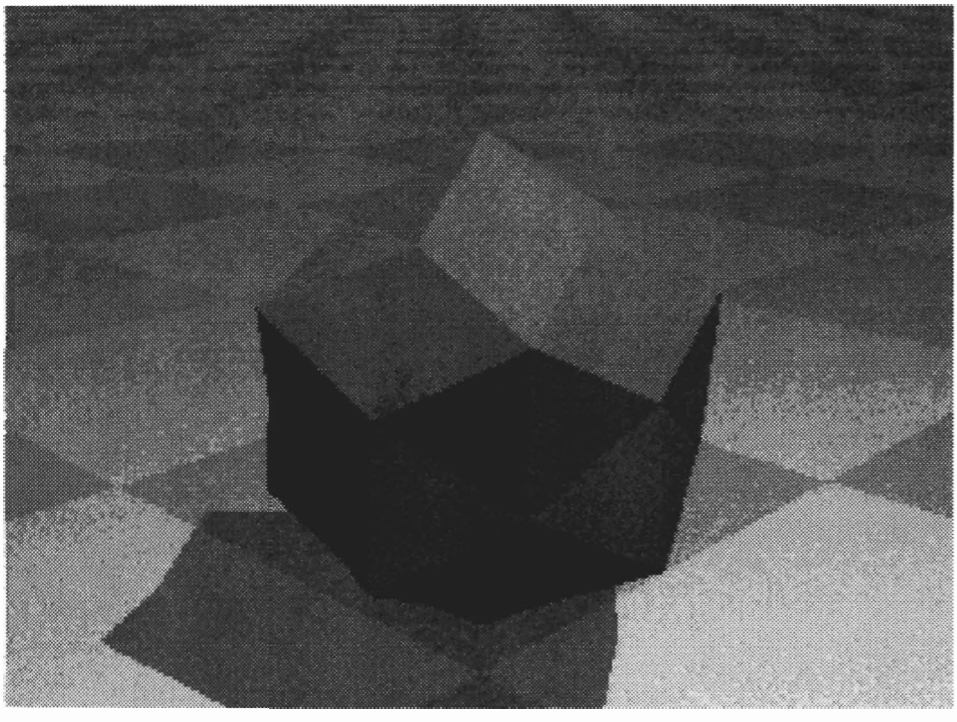
by

rl = r2 * (1.0 + 0.5 * sin(3 * a1)); /* 3 lumps, please */
dx = rl * cos(a2); dz = rl * sin(a2);

Dodecahedra

An interesting solid shape is formed when you bring together twelve pentagons. It is called a dodecahedron. When you set about trying to make a model of some new shape, it is a good idea to figure out some regularities that will enable you to calculate both the geometry and the topology.

If you were to set the dodecahedron on a table, one pentagon and hence a ring of five vertices would be at the bottom. Another ring of five vertices would be at the top. The remining 10 vertices lie on two rings, one a little bit above the center and the other below the center of the object. The upper two rings ar rotated with respect to the lower two rings by 36 degrees. Breaking out a pocket calculator and dim recollections of high school math quickly yielded the spacing and diameters of the rings. The geometry was established. The topology was no less obvious. In this case we resorted to drawing a dodecahedron and numbering the vertices in order to see which vertices to link together. The resulting code (on the following page) lacks elegance, as you can see:



Dodecahedron

```
makedodecahedron(object,properties)
struct object *object;
long int properties;
{struct vertex *v[20], *newvertex();
 double x,y,z,a,da,pi,c1,c2;
 double sin(),cos();
 int i,j;

 object->vertices=0; object->edges=0; object->faces=0;

 pi=3.1415927;
 c1=1.1755705; c2=1.618034;
 da=pi*0.4;

 z=0.5*(1.0+c2);  /* top ring */
 for (i=j=0,a=0.0; i<5; ++i,a=a+da)
    {x=cos(a); y=sin(a);
     v[j++]=object->vertices=newvertex(x, y, z, object->vertices);
    }

 z = 0.5*(c2-1.0);  /* upper middle ring */
 for (i=0, a=0.0; i<5; ++i, a=a+da)
    {x=c2*cos(a); y=c2*sin(a);
     v[j++]=object->vertices=newvertex(x, y, z, object->vertices);
    }

 z=-0.5*(c2-1.0);  /* lower middle ring */
 for (i=0, a=0.5*da; i<5; ++i, a=a+da)
    {x=c2*cos(a); y=c2*sin(a);
     v[j++]=object->vertices=newvertex(x, y, z, object->vertices);
    }

 z=-0.5*(1.0+c2);  /* bottom ring */
 for (i=0, a=0.5*da; i<5; ++i, a=a+da)
    {x=cos(a); y=sin(a);
     v[j++]=object->vertices=newvertex(x, y, z, object->vertices);
    }

 /* form the 12 pentagons */
 for (i=0; i<5; ++i)
    {link(object,v[i],v[(i+1)%5],properties);
     link(object,v[15+i],v[15+(i+1)%5],properties);
     link(object,v[i],v[i+5],properties);
     link(object,v[i+10],v[i+15],properties);
     link(object,v[i+5],v[i+10],properties);
     link(object,v[i+5],v[10+(i+4)%5],properties);
    }

 /* triangulate them */
 triangulate(object,v[0],v[1],v[2],v[3],v[4],properties);
 triangulate(object,v[15],v[16],v[17],v[18],v[19],properties);
 for (i=0; i<5; ++i)
    {triangulate(object, v[i], v[(i+1)%5],
        v[5+i], v[5+(i+1)%5], v[10+i], properties);
     triangulate(object, v[5+i], v[10+i],
        v[10+(i+4)%5], v[15+i], v[15+(i+4)%5], properties);
    }
}
```

After forming the twelve pentagons, they have to be triangulated. Recognizing which vertices belonged to which pentagon was achieved by resorting to the drawing of the dodecahedron. The triangulation could be done by adding two edges to each pentagon, but we chose to add a vertex at the center and then add five new edges, like this:

```
triangulate(object, v1, v2, v3, v4, v5, properties)
struct object *object;
struct vertex *v1,*v2,*v3,*v4,*v5;
long int properties;
{struct vertex *v,*newvertex();
 int i;
 double pos[3];

 for (i=0; i<3; ++i)
    pos[i] = (v1->pos[i]+v2->pos[i]+v3->pos[i]+v4->pos[i]+v5->pos[i])/5.0;
 v=object->vertices=newvertex(pos[0], pos[1], pos[2], object->vertices);
 link(object, v, v1, properties); link(object, v, v2, properties);
 link(object, v, v3, properties); link(object, v, v4, properties);
 link(object, v, v5, properties);
}
```

Finally, modify the main function to produce a dodecahedron:

```
main()
{struct object object;
 makedodecahedron(&object, 0x7070F000); /* pale blue */
 wrscene("dodecahedron.scene", &object);
}
```

The reason we chose an inefficient triangulation ploy was so that we could easily produce another solid shape, the stellated dodecahedron. If you take the last twelve vertices, the ones in the middle of the pentagons, and move them away from the center, then the dodecahedron will grow spikes. If you choose the expansion factor to be about 1.4721, then all the edges of the solid are equal and you have created a regular polyhedron other than the five known to Plato. He may have known of the stellated dodecahedron, and dismissed it as a cheat, because it lacks the perfect symmetry of his other regular solids. The stellation can be performed by the following function

```
stellate(object, factor)
struct object *object;
double factor;
{int i,j;
 struct vertex *v;

 for (i=0,v=object->vertices; i<12; ++i,v=v->next)
   for (j=0; j<3; ++j)
      v->pos[j] *= factor;
}
```

If you choose a large value for the factor you can produce very spiky objects, while values less than one will cause you to generate a dodecahedron with dimples.

Conclusion

By carefully considering the geometry and topology of objects, we have shown how you can write programs to generate solid models for your favorite rendering program. Now you can build anything from fractals to a body of water with waves.

InterComputing Inc.

2100 N. Hwy 360, Suite 2101
Grand Prairie, TX 75050-1015

Orders only call : 1-800-622-9177 (now incl. Texas)

Customer Service and Order Status please use
1-214-988-3500

Amiga is a trademark of Commodore-Amiga, Inc.

AMIGA

We are in business since 1984;
We support the AMIGA since 1985;
We will give you the best service possible at prices the other 'Discount-places' wont like - PERIOD !
Please call for a complete listing our products !
Member Better Business Bureau in Dallas

HARDWARE

Memory

8-Up Memorycard OK	145.75
8-Up Memorycard with 2MB	649.00
8-Up Pop-Simm 256k or 1MB	17.95
Starboard 0 MB A1000	239.75
Starboard 0.5MB A1000	399.75
Starboard 1 MB A1000	599.75
Starboard 0 MB A-500	239.75
Starboard 0.5MB A-500	399.75
Starboard 1 MB A-500	599.75
Starboard Upper Deck Ok	44.75

Drives

FData-10 ext. Drive	139.75
int. 3.5" Drive A2000 only	145.75
HardFrame SCSI Card	239.75
Starboard SCSI Module	97.75
Quantum 80MB Drive	875.00
Supra 20MB HDisk	649.00
Supra 30MB HDisk	749.00

Don't Pay More !

Flexible Data Systems

FDATA-10 Disk Drive

This external 3.5" Drive is a very inexpensive unit to make the life with your Amiga easier.
The FData-10 is compatible with the original Amiga-drive but is quieter, smaller and has a longer cable (27inch) to connect it to the Amiga.

Oh yes, before we forget to tell you, it's only $139.75 !

Video / Graphics

A-Pro Draw 12x12	499.75
A-Pro Draw 9x6	399.75
Amiga Live A 500	299.00
Amiga Live A 2000	329.00
Amiga Live A 1000	219.00
Digi View 3.0 NTSC	139.75
Digi View Adapter	19.75
Digi Droid	64.50
Easyl 500	299.75
Easyl 1000	349.00
Easyl 2000	349.00
Flicker Fixer	499.00
Flicker Master	12.50
Gen One Genlock	749.75
Imprint	399.75
Light Pen	99.75
Mitsubishi AUM 1371A	549.00
Panasonic 1410 Video Kit	259.75
Perfect Vision	175.00
Perfect Vision Color Splitter	85.75
Polaroid Palette	1995.00
Super Gen	699.75
X-Specs 3D	97.75

Great Hardware Buys

We now carry a wide variety of
MIDI-Equipment

Synthesizer	
Kawai K1 Synthesizer	749.50
Yamaha FB-01 Module	329.50
Percussions	
Alesis HR-16 Drum machine	379.50
Effects	
Alesis MIDIVerb II Digital Reverb	249.50
Mixer	
Kawai MX-8R Mixer (8 Channel / 2Effects I/O)	279.50
Midi	
ECE-Midi Interface	44.75
Gold Midi internal for A2000	69.95
Midi Cable	8.75

Sound / Midi

ECE-Midi w/ RS-232 thru	44.75
Future Sound Sampler	129.75
Soundscape Sampler	79.50
Soundscape Midi Interface	42.75
Perfect Sound A500/2000	59.75
Perfect Sound A1000	59.75
Midi Cable	8.75
Midi Gold for A2000 internal	69.95

Miscellaneous

Fuji 3.5" DS/DD Disks	17.75
Mouse Pad	6.95
Mouse Time for A1000	29.75
Multifunction Card	69.75
Phasar Gun	46.75
Processor Accelerator	149.50
Sony 3.5" DS/DD Disks	19.50
Supra Modem 2400 Baud	149.75

DESKTOP PUBLISHING

Comic Setter	66.75
Font Set I	24.75
Laserscript	29.75
Newsletter Fonts	29.75
PCM-Fonts Vol.1-3	27.50
Page Setter	89.75
Professional Page	239.75
Shakespeare	149.75
Studio Fonts	29.75
Zuma Fonts 1,2,3,4 each	24.75

MUSIC

Audio Master	39.75
Deluxe Music	59.75
Dr. T's 4-OP Deluxe	99.75
Dr. T's DX-Heaven	99.75
Dr. T's Kawai K1 Libr.	99.75
Dr. T's KCS 1.6	166.75
Drum Studio	29.75
Dynamic Drums	47.75
Dynamic Studio	129.75
Laser Sound Vol. 1-3	27.50
Midi Recording Studio	44.95
Music Mouse	49.75
Sonix	47.75
Sound Oasis	66.75
Sound Quest Casio CZ	84.75
Sound Quest D-110	99.75
Sound Quest D-50	99.75
Sound Quest DX-7	99.75
Sound Quest DX-7 II	117.75
Sound Quest Generic	84.75
Sound Quest MT-32	99.75
Sound Quest O-80-ESQ1	99.75
Sound Quest TX-81 C	84.75
Soundscape Pro Midi Studio	124.75
Soundscape Utilities I	39.95
Studio Magic	66.75
Synthia	66.75
Texture	99.75

Flight Pack

JET or FS-II
with 4 Scenery Disks
USA-East Coast
USA-North
Europe
Japan
$ 99.75

BUSINESS

Analyze 2.0	62.75
BEST Management System	299.75
Critics Choice	159.75
Cygnus ED	69.50
Excellence	189.75
Financial Plus	199.75
Flow	59.95
Gold Spell	29.75
Kindwords	66.75
Logistix	99.75
Maxi Plan 500	99.75
Maxi Plan Plus	129.75
Microfiche Filer	69.75
Microfiche Filer II	112.75
Money Mentor 2.0	59.75
Organize	66.75
Phasar 3.0	66.75
ProScript	32.95
Prowrite 2.0	72.50
Scribble 2.0	57.95
Softwood File II (GS)	79.75
Superbase Personal II	99.75
Superbase Professional	199.75
TexEd	22.75
TexEd Plus	54.75
Word Perfect Library	79.95
Wordperfect	197.75

GRAPHICS / VIDEO

3-Demon	69.75
Aegis Animator	82.50
Aegis Draw 2000	179.75
Analytic Art	39.75
Animate 3D	99.75
Animation Multiplane	56.75
Animator Apprentice	199.75
Butcher 2.0	21.75
Calligrapher new Version 1.05	87.75
Comic Setter	66.75
Deluxe Paint 2.0	94.75
Deluxe Photo Lab	99.75
Deluxe Productions	144.75
Deluxe Video 1.2	94.75
Director	44.95
Dynamic Cad	299.00
Fantavision	39.95
Flipside	39.75
Forms in Flight II	77.75
Gallery 3D	43.75
Interchange	29.75
Intro CAD 2.0	49.75
Kara Fonts	52.75
Lights, Camera, Action	52.75
Modeler 3D	72.50
Pageflipper Plus F/X	99.75
Photon Paint	66.75
Pixmate	49.75
Prism Plus	44.75
Pro Board	379.75
Pro Net	379.75
Pro Video Plus	199.75
Sculpt 3D	**69.75**
Sculpt 3D (PA)	**69.75**
Sculpt 4D	**399.75**
Silver	99.75
Terrain Disk for Turbo Silver	19.75
TV Show NTSC/PAL	62.95
TV-Text	62.95
Turbo Print	32.75
Turbo Silver	129.75
Video Effects 3D	129.75
Video Titler	99.75
Videoscope 3D	119.75
Window Print II	22.75
X-CAD	359.00
Zoetrope	99.75

GAMES

4th and Inches	29.75
Andromeda Mission	27.75
Amegas	19.75
Arazok's Tomb	24.75
Arkanoid	19.75
Balance of Power	33.75
Barbarian	24.75
Battle Chess	31.75
Black Shadow	25.75
Capone	29.75
Carrier Command	29.95
Centerfold Squares	22.50
Championship Basketball	29.75
Championship Baseball	27.75
Championship Football	29.75
Championship Golf	27.75
Corruption	27.95
Cube Master	19.75
Defender of the Crown	35.75
Destroyer	27.75
Ebonstar	29.75
Emerald Mine	17.75
European Scenery Disk	19.75
Faery Tale Adventure	29.75
Ferrari	32.75
Final Assault	31.75
Fire and Forget	28.75
Flightsimulator	39.75
Foot Man	22.75
Foundation Waste	27.75
Gee Bee Air Rally	27.75
Global Commander	29.95
Grand Slam (Tennis)	29.75
Guild of Thieves	29.75
Hollywood Poker	24.75
Hybris	27.75
Interceptor	34.75
Into the Eagles Nest	24.75
Jet	32.75
Jewels of Darkness	9.75
Karate Kid II	27.75
Knight Orc	29.75
Major Motion	27.75
Menace	21.95
Mindwalker	27.50
Obliterator	29.75
Offshore Warrior	28.75
Pioneer Plague	27.95
Plutos	19.75
Portal	34.75
Ports of Call	29.75
POW	25.75
Q-Ball	19.75
Realm of the Warlock	24.75
Return to Atlantis	34.75
Road Raider	27.75
Roger Rabbit	29.95
Scenery Disk 7/11	19.75
Scenery Disk Europe	19.75
Scenery Disk Japan	19.75
Shadowgate	29.75
Shanghai	27.75
Silent Service	27.75
Silicon Dreams	9.97
Sindbad	33.75
Skychase	27.75
Space Cutter	19.95
Space Quest	32.75
Starglider	29.75
Starglider II	29.75
Starwars	27.75
Street Cat	17.75
Street Sports	14.95
Strip Poker	27.75
Strip Poker Data #4 or #5	14.95
Surgeon	29.75
Terror Pods	29.75
Test Drive	29.75
The 3 Stooges	34.75
The Pawn	29.75
Torch 2081	19.75
Uninvited	33.75
Vampires Empires	29.75
Virus	19.75
Vyper	22.75
Wintergames	14.95
World Class Leaderboard	29.95
World Games	27.75
Zoom	21.75

UTILITIES

Anti-Virus IV	39.95
Arexx	32.75
CLI Mate	27.75
Disk Mechanic	56.75
DOS-2-DOS	42.75
Demonstrator	24.75
Encore	44.95
EZ-Backup	32.75
Facc II	19.75
Fine-Print	32.75
Marauder II	27.75
Project D	32.75
Quarterback	44.95
Raw Copy	39.95

BOOKS

AMIGA Graphic/Sound	17.75
Amiga Dos Manual	19.50
Desktop Video	14.95
Hardware Reference Man.	19.75
Intuition Reference Man.	19.75
ROM-KERNEL: Lib.&Dev.	27.75
ROM-KERNEL: Exec.	19.75

MISCELLANEOUS

A-Talk III	67.95
Intellitype	34.75
Online	44.75
The Works	129.75

New Software in Stock

4x4 Off Road Racing	Garrison II
Battleship	Micro Fiche Filer II
California Games	ProScript
Dragon's Lair	Pro Sound Designer
Dr. T's Kawai K1 Lib.	Sword of Sordan
Dungeon Master	Tetris
Falcon	TV-Sports Football
Gallery 3D	Zuma Fonts 4

PROGRAMMING - LANGUAGES

A/C Basic Compiler	119.95
A/C Fortran (AB-Soft)	179.75
Aztec C Developer	199.75
Aztec C Professional	145.75
Aztec Source Level Debug.	59.75
Benchmark Amiga Library	64.75
Benchmark C-Library	64.75
Benchmark IFF Library	64.75
Benchmark Modula-2	134.75
DSM MC68000 Disassembler	42.75
Lattice-C ++	395.00
Lint	69.00
MCC-Pascal	79.95
Modula-II Commercial	175.00
Modula-II Developer	89.75
Modula-II Standard	59.75
Power Windows 2.5	56.75
True Basic	69.75

Trade-UP!

We will Trade in your Amiga 1000 or A500 if you want to upgrade to the A2000 or the new A2500.
Please call for a Quote !

Go ahead !
Call us with your best Quote !
And as allways;
we've got the most
'customer friendly' Terms :
S/H $3.95
30$ Minimum Order
NO Credit Card Fee !
but
in TEXAS add 7% TAX !

All prices are subject to change without notice !

Animation Confidential

Secrets and techniques

by Derek Grime

Derek is the president of Beyond Graphics and began producing computer animation on the Amiga in October 1985. The studio has produced sequences for the CBS and CTV television networks as well as Commodore's broadcast commercials in 1987-88. He can be contacted at: Beyond Graphics, P.O. Box 1249, Stn. F, Toronto, Ontario, Canada, M4Y 2V8.

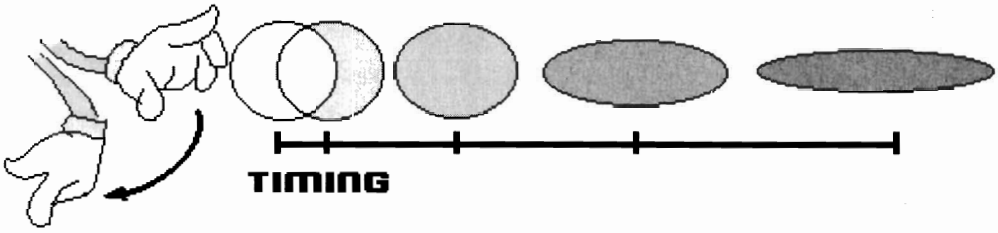
Bugs, Daffy, Donald and Mickey... some of Hollywood's most beloved stars have never left the second dimension.

These character's antics reached their summit in the earlier part of the century. Thousands of illustrations painstakingly hand drawn and painted are used in every production. So convincing is the illusion of life, we react to the characters on the screen as if they were actual flesh and blood. Quality such as this does not come at bargain prices. As labor costs grew and the public turned to television, cartoon animation slowly went into a decline. Unfortunately most of us see animation today as medium designed to sell toys to preschoolers. Things are only now beginning to change for the better. Contemporary motion pictures such as "Who Framed Roger Rabbit?" and "The Land Before Time" have signaled a resurgence in the popularity of true classical animation.

Computer animation started to gain popularity in the mid nineteen-seventies. Traditional animators first looked at these new tools with great interest. Although the graphics were attractive, the cost and ease of use of the systems were not. What started as a potential renaissance quickly became a morass of spinning chrome logos. In short order, the viewers became weary of all the eye candy introducing every show. There is the odd film that breaks the boredom barrier. If you've had the opportunity to enjoy "Luxo Jr." or "Red's Dream" you know that computer images can be just as effective as pencil and paper. The creative process is different but the emotional reaction from the audience remains the same. It appeared that classical and computer animation would combine in only the rarest of cases.

Exaggeration

Enter the Amiga, inexpensive yet powerful, with an artist interface that approaches the transparent. The Amiga is the natural path that the cartoon animator can take to adapt his skills to computer-created movement. Excellent peripherals are available to transfer illustrations to the Amiga, either with pressure sensitive tablets like Anakin's Easyl or digitizers like Newtek's Digiview. Page-flipping software sequences the many frames while the blitter takes care of the real time screen updates. Sprites, Bobs and bit-maps can be manipulated faster than on systems available for more than ten times the price.



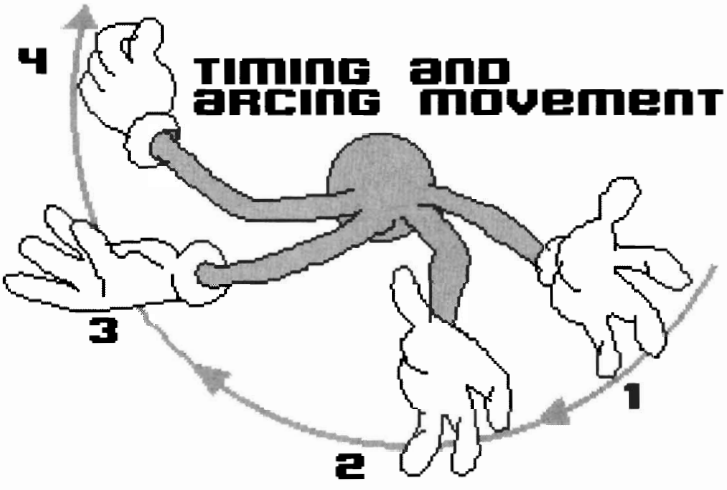
In the Amiga we have a nearly perfect classical animation tool. So why is the vast majority of Amiga animation so Godawful? Characters move around the screen like small robotroids, walking in jerky goosesteps and poking battle axes at each other as though they had the weight of drinking straws. You don't care if they live or die, because they never breathed in the first place. The animation rules that have been laid down over the past seventy years have been neglected. The Amiga's not to blame for these primitive animitrons, so let's do a little finger pointing.

I'd like to venture that animators guard their secrets with all the vigilance of a medieval guild. In the animation world the common estimate is that it takes seven years to complete the apprenticeship from illustrator to animator. Most of the technology we use has changed very little from the turn of the century. It's not uncommon to work weeks to produce a few seconds of "the good stuff". Techniques are passed from one generation to the next. In fact the theories I learned were unchanged since 1938. Thinking about it in these terms I suppose we are a medieval guild. And you know that guilds don't work cheap.

This leaves a bit of a dilemma for those of you working on Amiga software that just doesn't have the visual punch you had hoped for. You can hire a professional animator who hasn't got the old world fear of computers (that's a plug of course). More realistically, you can apply some of those time-honoured techniques of animation to your own project.

At the risk of being tossed out of the guild I'd like to dust off the massive tome animators keep with them at all times and translate some of these secrets to you from the original latin.

Exaggeration



Fire up the VCR and feed it some of the finest Warner Brother's animation from the early forties. Use the frame by frame mode to take a look at the detail that you never knew was there. Directors like Tex Avery and Bob Clampett were masters of exaggeration. Cartooning is caricature and everything must go one step beyond were you think it should stop. Bit maps should stretch and squash during the extremes of any action. If your character lands with a thud, it should flatten down to almost the ground level before it recovers. Instead of thinking in terms of robotic movement imagine the character as a rubber bag filled with jello. After all, that's closer to what we are.

Timing

Computers like to take a point from A to B and divide it into nice even increments. In the real world it takes time to gain velocity and time to lose it. Characters have to follow these rules as well as inanimate objects. Vehicles should slowly gain speed as they travel from place to place. The same applies to stopping. At the conclusion of the action, slow into the stop. Animators call this "slowing in and slowing out". As any stand-up comedian will tell you, timing makes the joke. In animation, timing makes the movement.

Arcing motion

Unlike machinery, nature cannot move easily in straight lines. Any action that we do calls upon a network of ball and swivel joints that prefer to move along curved paths. Swords don't jab in a perfect vector, they swing wide around the person throwing the blow. Try moving your head from side to side in a straight path. Feels pretty unnatural, doesn't it? Characters that move this way look unnatural too. The chin should touch the collar bone in the middle of the movement. Arms and legs travel in semicircles bending at the joints. This theory applies to three dimensional movement as well. Adding a dip in the middle will give you more life-like animation, even if the object is inorganic.

Anticipation, action, reaction

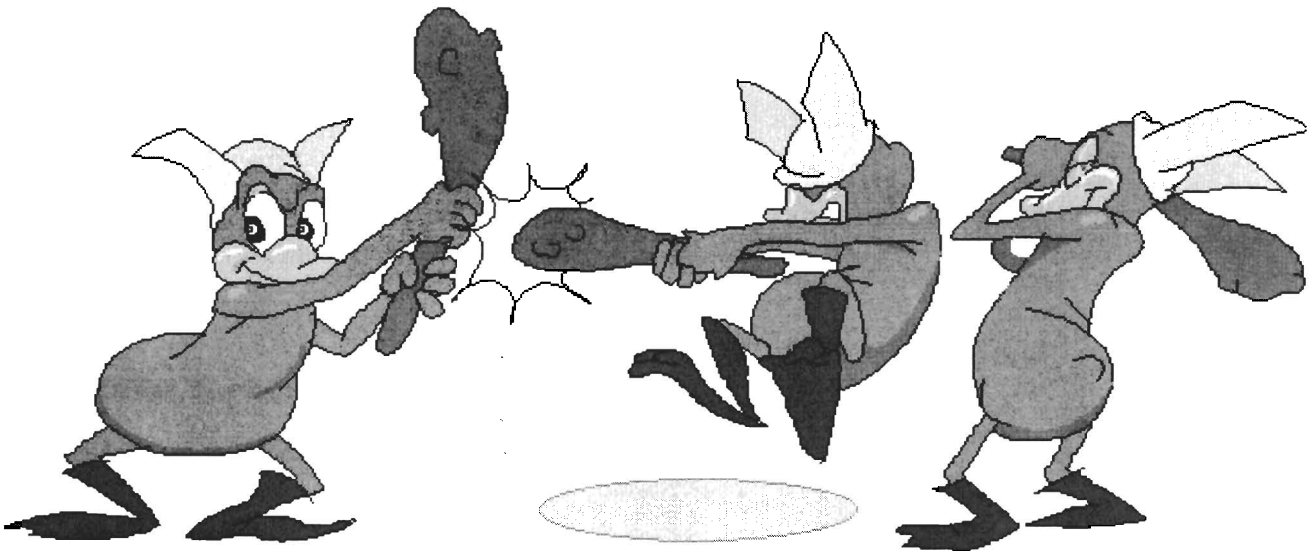
This is one of the more difficult of the classical animation concepts to understand. Every gesture, no matter how small, is divided into three parts. The move is first anticipated, giving visual clues to the viewer that something is about to happen. Secondly, the action itself takes place. Finally, there is a reaction: momentum is lost and things return to normal. As an example let's imagine a swordsman wielding his weapon. Many games take this situation and ignore all the possibilities, having the swordsman jab at his adversary in choppy little movements from the chest. While this is easy to animate, there is little emotion and no power in such a simple action. Here is how an animator would handle the same problem:

Anticipation: The swordsman first heaves his weapon over his shoulder. His eyes remain on his victim as he bobs the sword slightly up and down, feeling its weight. His feet are planted firmly, his stance resembles a batter at the plate, waiting for the pitch. He remains frozen, looking for an opening.

Action: The swing. Wincing with effort the death blow begins, velocity gaining in every frame. The arms are locked at full length and the blow travels in a huge arc outward from the body. At the apex the sword is closer to the viewer and should be drawn in forced perspective, bringing us into the action. What happens at the end of the swing is best left to the imagination...

Reaction: We're not finished yet. Some of the momentum is lost when it strikes the victim's body but there is still plenty of power left. Allow the sword to continue on its arc, pulling the character with it. The action should stop with the sword resting on the opposite shoulder, the body twisted from the shift in weight.

Keep in mind the three part move. Even less dynamic actions must follow these rules. The simple act of sitting at a table and reaching for a mug of grog is broken into threes. The charac-



ANTICIPATION -WIND UP

ACTION -SWING

REACTION -FOLLOW THROUGH

ter's hand first travels up and away from the mug, opening as it does. It holds for a moment, paused in mid air. Next the action itself as the drink is snatched from the table top in an arcing sweep of the arm. Finally the follow through as the mug is lifted up and away before heading for the mouth. Spill some of the grog during the lift and you're starting to think like an animator.

Characterization

Let's visit behind the closed doors of the typical studio. We see animators sitting at their light tables grimacing into small mirrors and gesturing wildly with their hands. One or two walk between the desks simulating lumbering monsters or hopping rabbits. This madness is called "getting into character". Good animation is good acting, and if it means tossing props around the room or doing facial histrionics, your finished work will look all the better. Keep a stopwatch nearby to record how long each action takes. Use it as a guide when deciding the number of frames. Remember that large characters move slowly and small characters will move faster and bounce higher. Lunatics like Daffy are loose and fast while "human" beings like Jessica Rabbit move slowly and distinctly. Machinery, robots and vehicles demand a great deal of precision while nature is free flowing and untamed. Study the classic features to learn what makes a great character tick. It's the type of homework that nobody minds doing.

Overlapping action

Part of the reaction stage of a motion is overlapping action. Simply stated, this is the extra simultaneous movements that occur at the same time as the main event. When a car's brakes are slammed to the floor, the passengers inside lose momentum at a different rate. Unfortunately they continue on after the vehicle stops. The velocity that's then checked by seat belts moves to the hair and loose articles of clothing. If a horse stops suddenly, its mane and reins whip forward. Thrusting a weapon outward requires that the body moves backwards.

Overlapping action is a lot of cause and effect and a little artistic guesswork. Often it makes the difference between the good, the bad and the ugly. One job I'm aware of where overlapping action was the answer involved a realistic head turn. Using a high-end 3D work station, the production team laboured to create a fully rendered photo-realistic human face. The results cost thousands of dollars and were visually stunning. Yet something was missing. The client wasn't happy. The move was just a little too stiff and unreal. Without the time or money to redo the sequence from scratch, an animator was asked to take a look at the problem. The solution was quite simple. A pair of large earrings were added to the character. As she swivelled her head the earrings flew outward. When she stopped turning, the earrings swung back and forth slightly before coming to rest. The new details were added at the post production stage, drawn onto the scene with a paintbox so that no extra rendering time was required. Overlapping action saved the day.

Shortcuts

There are no shortcuts. It's the extra touches and care that make us really suspend our disbelief.

Of course there are always strings attached. Better animation in your programs requires more effort. Even if you're willing to spend the extra time, your results may lack some grace for a while. Although at times this can be frustrating, practice will only make you better.

* * *

For a generation, users have tolerated fantastic games and adventures with poorly executed animation. The hardware isn't at fault. The Amiga can handle just about any creative problem you'd care to throw it. Following these time-honoured techniques will greatly improve the look of your creations. Computer animation doesn't have to be bad, it's just drawn that way

Online Lingo

Textual shortcuts, protocols, and smily faces

by James D. Williams

Jim Williams is an Amiga artist, animator and aspiring programmer. When not at his Amiga working on artwork for the local user group or learning C, he can be found hanging out in Compuserve's Amigaforum conference areas.

Let's face it; most computer users are lazy. But, a lot of us have a problem; we also like to talk. This is no problem in ordinary life (well, for *us*; those we impose our conversation on might have a different view on the matter); but, since we are computer users, we often like to talk using our computers. The Amiga encourages this practice, since we can be doing something useful (but boring) in the background while we talk to other computer users, or at least, leave messages on a BBS for other lazy computer users to read. Unfortunately, not many of us are trained typists/ semanticians. So, being true to our lazy nature, we take shortcuts.

If you've ever gotten on a BBS or involved in a conversation on one of the networks, you've almost certainly seen such cryptic messages as, 'IMHO', 'BRB', 'SYL', 'IC', and many others. It's not some secret code thought up by hackers to hide their activities from the general public and/or add to their mystique. No, basically, it's just an extension of some computer users' natural tendency to do as little work as possible.

The 'code' involved is very easy to decipher, since basically the messages are merely the first letters of a statement. 'IMHO', for example, translates to: 'In My Humble Opinion'. 'BRB', decrypts into: 'Be Right Back'. From this humble beginning, the rest of the abbreviations can be discovered from the context. If you see a 'BRB', typed in from one member of the online conference, or at the end of somebody's message on a BBS, and a reply somebody gives is a 'SYL', after a little thought you can come up with 'See You Later'. 'IC' can be uncovered merely by pronouncing it (again, assuming you know the context of the message): I See.

From here it becomes so obvious that it's almost boring. 'B-BL' stretches into 'Be Back Later', 'WHS' becomes, 'What He Said'. Of course, there's no set rule against making up your own abbreviations. One of the most commonly asked questions is; "Where are you calling from?" Des Moines, Iowa is a bit much to type for a lazy person like me so, it's become simply: Awa. Of course, the asker of the question might be a lazy type as well, in which case the question would look like 'WAYCF?' or even 'WAY' (though that could be confusing).

As I briefly touched on in the case of 'IC' and 'Awa', some of the changes to the language are phonetic in nature. For instance, 'Wanna go to AmiEXPO?'. Wanna is contracted from 'Want to', since it's not only fewer letters, the double 'N' in it makes it even faster and easier to type. 'Dint ICU there lst year?' is the lazy version of, 'Didn't I see you there last year?' Sometimes the online version is longer than the word it's replacing, such as, 'Naw, that was sumbuddy else.' The reason for this becomes obvious, when you think about the fact that there are no visual clues to follow as to what a message might *really* mean.

And that's a delicate and important matter. Pronounce a sentence one way, and it's a joke. Pronounce it just slightly differently, and it's an insult. If we were to only type perfect english, not only would the messages become bland and boring, but half the meanings would become lost. And so we come to the matter of online etiquette.

The rules of online etiquette are similar for both BBS's and on the networks, with the few obvious exceptions given the fact that the conversations on the networks are real-time (I refer to conferencing; the message bases of the network systems such as Compuserve are much like any other BBS). One should always announce oneself as soon as possible after entering a conference. Most experienced CO'ers (COnferencers, another modemism) will periodically do a **/sta** (STATUS: on most, if not all, networks, commands to the system start with a '/' as the first character on the line). This tells them how many people are in COnference. If the number goes up, they then do a **/us co** (on Compuserve), or a **/list** (on People/Link) to find the ID's of everyone in COnference. Sitting on your thumbs and just reading what others type is called, 'Lurking', and is considered rude on the networks *and* on BBS's.

There are many ways to announce your presence. Some simply type, 'Hi!', or 'Hello!'. Some send cute little messages, usually programmed into their terminal software as macros. One fella on Compuserve always lets you know he's arrived by sending, 'Klunk!'. Me, I always announce my presence by typing, '>----- POP! ----<'. Not only does this make it stand out

that a new person has arrived, but, by using the same greeting, pretty soon people get to know who it is that just arrived.

Some network forums and BBSs require that you use your full name; however, in most cases, you can use pseudonyms. If you are required to use your full name, just use your first name in conference, since most names can take up a good third of an output line and distract terribly from your message. If you're on a BBS that requires a full name, sign each of your messages with the name or nickname that you like to be called (on the networks, your name can be set by typing the command: **/nam** *name*).

You should also announce your comings and goings. You can either spell out the reason ("I gotta go potty!"), or simply make do with BBIAB, BRB, BBL, and so on. The former adds to the feeling of being in the same 'room' as the others, whereas the latter is very appropriate whenever you have an emergency to attend to.

On the other side, whenever somebody announces his arrival, you should always acknowledge his presence before continuing your conversation. A simple 'hi' will do. It's especially important for new people, since they probably feel intimidated already, and feel as though their statements are interrupting something important. If you go and return, let the others know that you are back. The networks keep track of who's in a given conference room but they can't tell if you are at your terminal or not. No one will know whether you're back or not if you don't tell them. You might also be working on something in the background (a common occurence, thanks to the Amiga's multi-tasking!) A simple IB ("I'm Back") would suffice. As a matter of fact, while you're busy with something outside of CO, you might want to add to your name, 'Busy', or 'BRB'. If someone comes back into CO, you can acknowledge his reappearance by a simple, 're'. Here's an example:

(1,Dougie)<25> When d'yall think WB1.3 will REALLY be out?
(1,Larry)<16> When it's out.
(1,Eet)<98> RSN, Dougie.
(1,Keith)<64> Hi, All!
(1,Larry)<16> hello, Keith.
(1,Dougie)<25> Hiya, Keith!
(1,Eet)<98> Acc! Complaint Dept!
(1,Keith)<64> What is it now, Eet?
(1,Dougie)<25> BRB
(1,Eet)<98> later, Dougie.
(1,Larry)<16> Anyway, I just thought I'd bring up Arexx again.
(1,Dougie)<25> IB
(1,Keith)<64> reDougie
(1,Larry)<16> reD
(1,Eet)<98> reDougie.

The 'reDougie', or 'reD' means, basically, 'hello again'.

As I said earlier, typing in messages lacks certain visual clues that makes conversations (either real-time or on a message base) more interesting and fun to participate in. It also allows ambiguities to creep in and possibly cause some antagonism. That's why adding to what one types is very helpful. If you make a joke, like Eet did above (Acc! complaint Dept!), the recipient might be insulted rather than amused (as Keith might have been, since he wrote Access.) To avoid hurt feelings, some symbol of humour should be added. The most common is the <grin>. Most asides are placed in angle brackets, since they usually describe some physical action taking place <inserting peanut butter sandwich in modem>. There are other delimiters possible (such as ::coffee:: to announce that you're going after some coffee), but, angle brackets are somewhat of a standard.

You're not restricted to <grin>s to express amusement, either. <chuckle>, <guffaw>, <snort> all give the nuances of humor that body language would in a face to face conversation. If someone is eating potato chips while CO-ing, for example, he might type '::munch, munch::' once in awhile. If someone says something really funny, there are several ways to laugh at it; heheh is sort of a small chuckle, heehee is a giggle, har! would be a belly-laugh, something really laughable might get a hahahahahahahahahahahaha! across the screen.

You aren't restricted to showing humor, either. If you want emphasis, use capital letters. NOTE: CAPITAL LETTERS ARE USED TO INDICATE THAT YOU ARE SHOUTING, AND IT'S IMPOLITE TO SHOUT IN CONFERENCE EXCEPT WHEN YOU MEAN SPECIAL EMPHASIS ON SOMETHING. CONSTANTLY TYPING IN CAPITALS WILL PROBABLY HAVE EVERYBODY TYPE: **/squ** *yourname*, which would mean that they wouldn't be able to read what you type (/squ stands for /squelch).

Being Amigoids, we're both lazy and creative. So, sometimes graphic images are used to express emotion. **:)** is a smiley face on its side. **:^)** gives it a nose. There are endless variations, and, the emotion generally depends on the direction of the mouth: **:-)** means happy; **:-(** means sad. **8)** means crazy, **B)** means cool (wearing sunglasses). **:P** means you're sticking you're tongue out (the written equivalent is Phthpppppp!), **8D** is a big happy grin. **;)** is a wink, **:^|** means no humor.

There's also specialized forms of talk. Often, since commands begin with a /, you'll see messages such as: **/nolist** or, **/noCtalk**. These are made by putting a space before the /. Say, if I ask somebody if they want me to list the source code to my great display hack, and you don't want to watch my code list down the screen for the next few minutes, typing that as soon as I ask lets me know that at least *one* person doesn't want to see it. Other cases of specialized talk are when we speak in different languages, for example:

#define Dougie Macintosh_user

is C language for saying that Dougie is a sub-amigoid.

I think this gives you a pretty good basis for communicating with the odd creatures that inhabit the networks and BBS's. Don't be afraid to say, <grin> <snort> :) <kerchuckle>.

Introducing ROBBS

An ARexx application for serial communications

by Larry Phillips
Copyright © 1988 Larry Phillips

The Amiga, by its very nature, encourages a different sort of program. On some machines, you will see word processors integrated with database managers, spreadsheets, and mailing list programs. Great efforts are taken by the authors of these programs to provide a lot of features, and to smoothly integrate the various parts of the package. You will also, on single-tasking computers, find features in large and small programs that do mundane small jobs, like deleting a file, listing a directory, and so on. The reason these features are there is because it is usually not possible to do the job unless you are willing to exit from the program. If you've ever looked for a particular program, or set of programs, to do something like remote updating of a database over the modem, you have probably come away frustrated that the only programs you could find were those that could only do it in a 'batch mode', after the day's business, and while your local database program was inactive. Well, we Amiga owners know that there is a better way, and we have the machine that can solve this type of problem.

Because the Amiga is a multi-tasking machine, there is not much reason for an author to include those features that are best handled by other programs. As an example, your terminal program does not need to be able to get a directory listing, because it is just as easy to flip to a CLI (or bring up another CLI) to do the job. In effect, your simplest terminal program is far more powerful, or at least potentially far more powerful than any you might find on some other machines. You can run virtually all other programs while online, giving you a freedom and flexibility that can only be dreamed of on lesser machines.

While this ability is important, there is another level of versatility that is, as yet, largely untapped. It is nice to be able to run different programs at the same time, but wouldn't it be still nicer if you could use any program with any other program as if they had been *written* to work together? It isn't possible today, at least not with all combinations of the programs we have available, but it could be at some future time. I have heard an Amiga owner say, "A database, word processor, and animator linked together... could put Hypercard to shame." Personally, I can't think of a single use for those three program types acting in an integrated manner, but the fact is that someone did think it would be a marvellous thing. I doubt the author of any database program would even consider writing the code to allow it to link up with the other two, and the same goes for the other two authors. Even if they did think it was a good idea, what programs would they choose to link up with? If it happened to be one you particularly like, you'd be happy, but if not, you would probably keep wishing that he'd chosen your favourite.

These thoughts led me to the idea that what we really needed was a series of building blocks designed to be hooked together, each providing a limited, but necessary set of functions. I can't take credit for the basic idea though. This is a philosophy which has stood the test of time in at least a few operating systems. Unix is one, and even if you have no particular love for Unix, you must admit that the ability to build complex programs from simple, small programs is a powerful idea. I wanted to go one step further though, and write a series of modules that can be controlled to a finer degree. While Unix allows manipulation of programs via stdin, stdout, pipes, redirection, and intermediate files, I wanted to 'get into the guts' of small programs and allow them to be manipulated more directly.

The Amiga has a built-in feature that is ideal for our purposes. It can send messages between programs via a system of message ports. A message port is a simple idea. A program can create a message port, give it a name, and register it with the operating system. Once that is done, any other program can look for the message port by name; if the name is found, the program can then send a message to the port. The message is received at the message port, and the program that created the port can act upon the message in any way. The receiving program, of course, must be able to understand the message - that is, it must have knowledge of the format in which the mes-

sage's data is stored. The receiving program then replies to the message, to let the sending program know that it was received, allowing the sending program to re-use the memory allocated for the message.

Prior to the advent of ARexx, the messaging ability of the Amiga was readily accessible only to programs written in assembler, or in compiled languages like C or Modula-2. This had the effect of putting the capability beyond the reach of many Amiga owners.

ARexx, however, is an interpreted language with full messaging ability. It can send messages to other ARexx programs (scripts) or to programs written to understand ARexx messages. This opens up a whole new area for those would-be programmers who like interpreted languages, allowing them to write simple (or not so simple) ARexx programs to call other programs as 'building blocks'. The result is a sort of a computer 'erector set' for applications.

After months of talking to people about this idea, and not being able to interest anyone in it to a sufficient degree thet they would actually write modules, I decided to write a few myself. I have called the series of modules 'ROBBS', for *Rexx Object Building Block System*. The first module in the series is called 'SerMod' (for *Serial Module*). It was designed to be small (it's currently about 12 kilobytes, but it could be made smaller still), and to have a limited set of functions, dealing only with things that need to be done for serial communication. It started life as a combination serial module and display, but as soon as the serial part was working to my satisfaction, I split the display out and started the second module, called (you guessed it), DispMod. DispMod was a simple display program having only one type of window, and only enough functionality to be able to handle keyboard input and text output.

By itself, SerMod is somewhat limited. You can run it from a script, and send it commands from the script to do a variety of jobs, such as setting the serial port parameters, sending text, receiving text, and checking for matches with user defined strings on the incoming data. With an appropriate script, it could act as a file transfer utility, a BBS, or a program to allow remote CLI-like operation of your machine (in a limited manner). With the addition of DispMod, the combination of a script, SerMod, and DispMod become a fairly powerful terminal program. When I say 'fairly powerful', I refer to the complete scripting capability provided by ARexx, since the protocol modules, menu module, gadget module, and so forth, have yet to be written.

I chose to write SerMod first, since many programs can benefit from serial communication capability, and because it is not difficult to visualize ways in which you can combine a serial module with other programs. SuperBase is about to arrive (it may have already), with an ARexx interface, MicroFiche Filer already has one. CygnusEd Professional, TxEd, UEdit, and DME all have ARexx interfaces. PCLO and PCLO Plus have one. With SerMod and an ARexx program, all these programs have suddenly acquired the ability to communicate over the serial port, locally via a cable, or remotely via modem. If there is enough interest in this project, there will undoubtedly be more modules written, small programs that do only a few things, and that allow you to custom-build other applications. Even those who have no desire to program, even in ARexx, can benefit from the ARexx programs written for ROBBS.

Perhaps a few words about the philosophy behind ROBBS modules would be in order. A typical ROBBS application will consist of one or more ROBBS modules, controlled by one or more ARexx scripts. In addition, other programs might be called, whether or not they actually have an ARexx interface. Any program that can give you the data you need, perhaps in a file, can be used with an application.

Each module in a ROBBS application should do only a few operations, to keep the size down, to simplify programming, and to ensure that there is little duplication of effort between modules. Only those commands that make sense to the particular application need be provided. There need be no 'standard' set of commands across all modules, for the above reason, and because it is easy enough to change the controlling ARexx program to account for any differences. There are a few commands that make sense to all modules, and these should be called by the same name in all modules. Examples of these are STATUS, CONNECT, CTRL, and DIE (more about these later).

At this time, I'd like to give you an overview of what a typical module might look like from an ARexx programmer's point of view, and I'll use SerMod as an example. Though it is 'finished', it is not 'cast in stone'. If you have any suggestions for additional features, I would like to hear about them.

Commands that SerMod understands

RXD: Turns the serial received data on or off. This is to allow other programs that open the serial port in shared mode to have access to it without interference. RXD accepts 'ON' or 'OFF' as arguments.

TXD: Enables or disables transmitted data. This allows another program to send data without any interference from Sermod. The other program must open the serial device in shared mode. TXD accepts 'ON' or 'OFF' as arguments

SEND: Sends a string out the serial port. A variation of this command, LSEND, sends a string with an appended Carriage Return character. SEND and LSEND' accept a string as an argument.

MATCH: Sets up a 'match string' for Sermod to watch for in the received data. When a match is detected by SerMod, a message is sent to the CTRL port, specifying the number of the string that matched. Sermod allows ten strings of up to 100 characters each. The MATCH command can be issued any number of times, which means that you can replace any match strings already set up. MATCH also lets you clear any match string, or clear all match strings. Note that SCAN (discussed be-

low) must be active in order for a match to have any effect. MATCH accepts a number between 0 and 9, followed by a string, as an argument. Example: MATCH 3 'This is a string'

SCAN: Enables the matching of received data with the MATCH strings currently active. SCAN takes ON or OFF as a parameter.

CONNECT: Connects the received data with a named message port. CONNECT also lets you set up the command that will be used to send the data, since not all modules have the same command for receiving data. Sometimes you might want the controlling script to receive all incoming data, while at other times you would want the data to go directly to another ROBBS module. CONNECT accepts two arguments, a port name and a command.

CTRL: Connects Sermod to another port for control purposes. While any script program may send messages to Sermod, Sermod sends received data to the 'CONNECT' port, and 'matched' messages to the controlling port. This means that you can pass control from script to script, as you see fit. CTRL accepts a port name as an argument.

COMM: Lets you set baud rate, mode (echo, noecho), handshaking, and translation (7 or 8 bits for received data). The format for COMM's arguments is more complex than with the other commands; it is described in the detailed documentation accompanying the Sermod module, which will appear next issue.

DIE: Commands Sermod to quit and unload from the system. If all messages sent from Sermod have been replied to, this will be done immediately. If there are any outstanding unreplied messages, a message to that effect will be printed to the CLI from which Sermod was launched; you must then issue the REALLY_DIE message to remove Sermod. This should never happen in a debugged application, and is provided as an emergency escape for the times when a script terminates early due to an error. The command needs no arguments.

STATUS: Returns a string describing the current status of Sermod. It contains information on serial port settings, the SCAN, TXD and RXD states, the CONNECT and CTRL ports and, for future use, the address of the IOSerial Request structure Sermod is using. STATUS needs no arguments; the format of the returned status string is explained will be described in detail next issue.

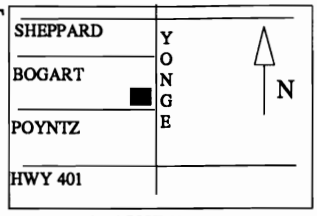
More to come...

Next month, we'll look at the source code for SerMod, a few example ARexx programs, and a more detailed explanation of the commands and how to use SerMod. Meanwhile, ponder what I have said here, particularly with respect to the versatility of hooking programs up through ARexx, and let your imagination run free. In the months to come, I will have at least one or two more modules, perhaps more, to present, and soon we will be well on our way to some of the niftiest applications you can imagine.

CANADA'S BEST SELECTION OF
AMIGA - C64 - C128 SOFTWARE
COMSPEC HARD DRIVES AND BERNOULLI DRIVES
AVAILABLE FOR
THE AMIGA 500 AND 2000

Electronics 2001
COMPUTER PRODUCT CENTRE

4736 YONGE STREET
WILLOWDALE ,ONT
M2N 5M6
TEL: (416) 229-2700



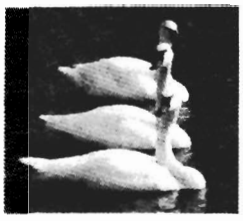
AUTHORIZED COMMODORE DEALER SINCE 1979

MCP ASSOCIATES, INC.
New York's Premier AMIGA Dealer
Computer Systems/Hardware/Software
DISKS (Blank Media): Repairs/Service
Competitive Pricing — Please Write/Call for Catalog
A2000HD The New AMIGA's A2500
SPECIAL: 68020 Board w/2MB — $1399.95
Shipping/Handling — Please Call
DISTRIBUTORS AMIGA SPECIALISTS
28-21 Astoria Blvd, Astoria, N.Y. 11102-1933
Tel: 1-212-221-6272 Fax 1-718-956-9028
Tel: 1-718-956-9000

CygnusEd Professional
by CygnusSoft Software. Published by ASDG Inc.

"The New King of Editors,"
68000er, February 1988

"This editor is definitely one of the best I've used on any machine, and I've used many."
The Amiga Sentry, Sept. 1988



CygnusEd Professional brings unparalleled levels of flexibility and speed to the **Amiga.** features include:

Crash Recovery. Variable screen resolution up to 1000 by 800. Full Macro capability. Over 200 functions and variables available through AREXX. Hot Key operation. Screen refresh at over 30,000 cps. Search and replace at over 100,000 cps. Blitter based horizontal and vertical scrolling. Vertical and Horizontal block cut and paste. Many Word Processing features. Many Programmer features. Edit multiple files on screen at same time.

Asynchronous Printer support. Super interprocess communications facilities. DOS and AREXX commands can be launched from inside the editor. User definable "bookmarks". International keyboard and video support. Mouse or keyboard based Turbo Scrolling. Keyboard and mouse can be interchanged for everything except entering text. Will create icons for WorkBench users. Edits binary files. And **MUCH more!**

ASDG INCORPORATED
925 Stewart Street
Madison, WI 53713
(608) 273-6585

Available At Your Local Dealer $99.95 U.S. Dollars

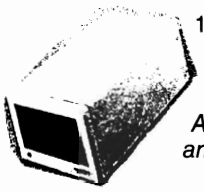
CygnusEd Professional Copyright 1988 By CygnusSoft Software. Amiga and Workbench are trademarks of Commodore-Amiga Inc
Developers are encouraged to contact ASDG for possible publication of your products

DR. OXIDE SLICES PRICES!

Comp-U-Save's cost-cutting clinician is at it again! This month Dr. Oxide, a bargain-basement surgeon if ever there was one, offers the industry's lowest prices on hardware and software, plus special deals on products available only through Comp-U-Save!

BUILD YOUR OWN HARD DRIVE KITS!

5¼" powered hard drive chassis with fan
3½" powered hard drive chassis with fan
only $120.00 each



1000 or 500 SCSI interfaces with passthru and software for
$145.00

All you need is a SCSI hard drive and you're ready to Rock and Roll!

Call for SCSI hard drive prices

AMIGA PUBLIC DOMAIN

OVER 700 DISKS!

Largest Amiga PD Library in the World
also C-64 & C-128 – Write for Free Catalogue
Amiga PD – $4.00 each

BUSEXPANDER FROM BILL'S BOARDS

The only board for the Amiga 500 or 1000 that expands either machine to 12 slots! Fits in any baby AT case and provides 6 slots for the 2000, 6 for the PC (4 of those for the AT). Now you can use most of the expansion cards designed for the 2000 – hard disk controllers, 2/4/8 meg RAM cards, A2088 Bridgeboard, etc. Use low cost IBM-compatible expansion cards already supporting a wide range of business and scientific applications. Designed to work with auto configurable cards. Meets ZorroBus and Amiga 2000 Bus electrical specifications. Available exclusively through Comp-U-Save!

12 Slots for your 500/1000! Price: $495

Amiga Hard Drives 500 - 1000 - 2000

20 Meg - $585.00 32 Meg - $699.99 48 Meg - $799.99

Amiga Dual Drive 500 - 1000 - 2000

With Own Power Supply – $399.00

Amiga External Drive $139.99

Only Uses Half the Power of 1010 – with Pass Thru Disk Drive & Monitor Extension
Cables 30"– $19.99

Panasonic WV1410 Video Cameras

For Digitizers - $199.99 16MM Lens – $29.99

Special 2400 Baud Modem – $154.00

★ Trackball	$45.00
★ Plastic Diskbank (Holds 120 3.5 in. Disks)	$16.99
★ Copy Arm (Heavy Duty)	$29.99
★ Mouse Mat (Fabric)	$ 5.00
★ Mouse Mat (Teflon)	$11.00
★ Gender Changers – All Types	Call
★ Static Mat (23.5 x 25.5 in.)	$24.00
★ Rapid Fire Joysticks	$12.00
★ Printer Buffer (32K–512K)	Call
★ RF Modulator	$14.99
★ A/B Switch (Ser.)	$13.99
★ A/B Switch (Par.)	$14.99
★ A/B/D/E Switch	$29.99
★ Crossover Box	$39.99
★ Cables 500 – 1000 – 2000	Call
★ Teak Diskbank (holds 150 3.5 in. disks)	$39.99
★ Teak Diskbank (holds 200 5.25 in. disks)	$39.99
★ The Library (holds 80 3.5 in. disks)	$19.99
★ Floppy Wallets (Many Sizes)	Call
★ 3.5 in DS/DD Disks (Bulk)	$1.10 ea.
★ 5.25 in. DS/DD Disks (Bulk)	$ .39 ea.
★ Books – All Titles 15% Off	Call
★ Memory 512K – 4 Megs	Call

Call for Details

Comp-U-Save

Come See Dr. Oxide
In Our Booth At All
World Of Commodore Shows!

410 Maple Avenue, Westbury, NY 11590
In NY State (516) 997-6707 (Tech Support) Outside NY State (800) 356-9997 (Orders Only) Fax (516)334-3091

A Device Is A Library Is A ...

An introduction to Amiga devices

by Betty Clay
Copyright © 1988 Betty Clay

Betty Clay, a former mathematics teacher, chose writing for her second career. She is the Amiga columnist on STARTEXT (a videotext service of the Ft. Worth Star-Telegram. She has a regular column with ICPUG, and her work has also appeared in The Amigan. Betty can be reached via CompuServe' (76702,337), QuantumLink (bjc), or by mail at 1322 South Oak Street, Arlington, Texas, 76010.

Devices are of vital importance to the operation of the Amiga. The trackdisk.device controls our disk drives, the Audio.device lets the Amiga speak, and so on. There are other kinds of devices too. When we ask for "Info", we may get a list of a dozen or more devices that are active in the system. My own system currently shows the following devices: Pipe:, Aux:, Speak:, Newcon:, VDK:, DF2:, df1:, df0:, Prt:, Par:, Ser:, Raw:, Con:, and Ram:. The names are familiar, and the general purpose of each is well-known. But of what, exactly, does a device consist? And what does it do?

It is not my purpose here to tell an advanced programmer how to write a new Amiga device, but to clarify the concept for myself and for other "beginner to average" users. Eugene Mortimore has filled an entire book with information about devices, and those wishing to write their own should consult it or another appropriate source (*Amiga Programmer's Handbook, Volume II*, Eugene P. Mortimore, Sybex).

The term "device" is used in at least two different ways. We sometimes speak of hardware devices, such as a disk drive or the serial port. Again, we may speak of those physical devices as being "Units" controlled by a device. In the Amiga environment, a device is the software that controls the physical devices. An Amiga device will always control one or more physical devices, called units. But even here, we use "Unit" to mean both hardware and software. Confusing!

When I asked the experts (consulted all of the available books on the Amiga), I found differing definitions of devices. All agreed that the device was a software concept, and seemed to agree that a device is a superset of an Amiga library. Libraries are more easily understood and are better documented, so it is easier to answer the next question, "What is a library?"

Amiga libraries

A library (more specifically, a "shared library"), is essentially a collection of routines that may be accessed by any Amiga task. The purpose of shared libraries is to provide programmers with standardized, often-used routines that would otherwise need to be included as part of the program code itself.

Each of the Amiga libraries consists of three main divisions: a Node structure, used to link it into the system; a set of jump instructions, giving the exact location of its various routines; and a library data segment. The address of the library is considered to be the first byte of the library node. The jump table vectors are located at negative offsets from this base address and the data items are at positive offsets.

The library node contains these items:

- A List Node, to tie the library to the rest of the system
- 1 byte to hold the flags that tell whether the library is valid, has a proper checksum, etc.
- 1 word (16 bits) to indicate the size of the jump table
- 1 word to give the size of the data table
- 1 word to give the version number of the library
- 1 word to give the revision number
- A pointer to a library ID string of up to 255 characters
- 32 bits to hold a checksum
- 1 word to hold the number of tasks currently using the library

The list Node structure is the same for all nodes in the system, and is familiar to anyone who has attempted to program the Amiga. Everything that must be kept in a system list begins with a similar list node. It consists of a pointer to the next node in the list, and one to the previous node, a byte identifying the type of data (device, font, interrupt, library, etc.), a byte that gives the node priority (which may be used for ordering the nodes in a List), and a pointer to the name of the node. The list node is not really a part of the library itself, but is just a means of keeping track of the things in the system.

The last item in the library structure, lib_OpenCnt, is incremented each time a task or process opens the library, and is

decremented each time the library is closed. If no one is using the library, it can be expunged from memory, and this will happen if the machine is running low on free memory. The incrementing and decrementing happen automatically as the tasks open and close the library, and programmers need not take further notice of it, except to be sure that they do close the library when they have finished using it.

When a program calls for the library (with the Exec library's OpenLibrary function), the system list is searched to see if the library is already in memory. If so, the open count is incremented and the program is given the library address. If it is not already in the list, the library will first be loaded, if it is available, and added to the library list. The location of the library in memory may be different for each version of the system software. For disk-based libraries, it may be different each time the library is loaded. In both cases, though, the jump instructions by which the library's routines are accessed will always be at the same (negative) offsets from the library's address. That is how software can remain compatible between different machines, or between different versions of the software. If a library routine is called in the prescribed manner, the system can find it no matter where it currently resides.

How does a program know the offset of a specific function, when the system only gives it the address of the library? That is where the compiler and linker fit into the picture. When the programmer links a program's object files to create an executable program, the linker supplies the offset into the jump table for each library routine invoked, and codes the offset into the program. (The linker, in turn, extracts the offset information from a link library; the standard one is "amiga.lib".)

The data portion of a library can be accessed by applications, but is more often used by the routines within the library itself.

Devices

It was a goal of the Amiga's designers to make every part of the system as consistent as possible. Consistency makes for easy programming. In the case of devices, however, it is a notoriously difficult goal to achieve. A light pen simply does not need the same commands as those used by the disk drive. The timer device require instructions that would only confuse the audio device. The devices were made as nearly consistent as possible, but each requires routines peculiar to its purpose.

Since a device is a superset of a library, we know that it must have a device library node, a jump table to the device functions, and a data table, and that these must follow the library pattern of having the jump table use negative offsets and the data area use positive ones. There are descriptions of the individual devices in the *ROM Kernel Manual* that describe the functions peculiar to each particular one, but there are also a few standard routines that are used by all.

The jump table for every device will have calls for Close, Open, Expunge and Init, and must allow space for three reserved functions. It must also have entries for BeginIO and AbortIO. These may also be standard commands such as Open, Read, Write, and Close, and there may be device-specific commands for such purposes as allocating a sound channel on the audio device.

Every device must be able to deal with related software structures called Units. The trackdisk.device is an obvious example, since we might have many drives managed by the one device. There would be an associated unit for each of the drives. This Unit is not hardware, however. It is a software structure that enables it to pass messages and data to and from the task or process that needs data.

Since the major function of any device is to manage input and output, understanding of the I/O functions is necessary to understanding a device. When a task opens a device unit with the OpenDevice call, the system automatically initializes the device structure and also a unit structure that will serve as a message port for the particular unit selected. The same device and unit can be shared by any other tasks that are running, and the OpenCnt slot in the device structure will keep a record of the number of tasks currently using it.

A task will use an I/O request structure that describes its needs for data, and will send this request to the message port of the Unit structure. The message will describe the data needed, and will tell the device where to put the data in memory. Normally, these requests are handled on a first in first out basis, but in some instances the task ask for "QuickIO", which may allow it to bypass standing in line.

Why?

The computers I used in the past were simpler machines, and made no use of libraries. Devices were strictly hardware to those machines. The Amiga's system of software libraries and devices is more flexible, and makes better use of memory. The code can be stored wherever in memory it fits, and the system lists keep it available. It is easy to upgrade the system, since the libraries and devices are just jump tables; relocating the actual routine does not affect user software. The order of the entries in the jump table remains constant, and a specific entry is found through indirect addressing. The linker allows the programmer to deal with the device and its routines by name, with no need to memorize lists of locations, as was the case with the older, absolute address machines.

This approach to libraries and devices is especially important in a multitasking computer. Once a library or device has been loaded into memory, it can be used by all tasks that are running. Each individual program can be much smaller because it does not have to contain coding for all of these functions. Since all tasks can share the same coding for functions that every task needs, less memory is used by having one copy of the code for all tasks to access. And finally, it is far easier to write a program when these common tasks have been written for you, and you need only tell the machine to do them.

Amiga Modula-2 Programming

Static variables, EXPORT QUALIFIED, CASE vs. ELSIF...

by B. Clark 'Top Gun' Williams

Clark Williams has been programming for 19 years in everything from COBOL and RPG to 68000 assembler, on IBMs using OS360, everything DECish, and even PICK systems. At last count he spoke 123 different computer languages (not counting dialects of Pascal, FORTRASH, etc.)... but his first love is flying. When not aloft, he writes Modula-2 programs.

Last issue we talked about 'Visual Acuity' and had an introduction to some structured housekeeping. This time, let's start with a look at static variables.

When does a variable get a value? When does the code (if it exists) in the main body of a DEFINITION MODULE get executed? The creator of Modula-2, Niklaus Wirth (pronounced "Virt"), doesn't state a methodology for module initialization. TDI's Modula-2 does give us a general rule of thumb that modules are initialized in order of declaration, but doesn't give us a guarantee that this is the execution order. We are told, however, that one can be reasonable sure that the module will be initialized prior to the first module invocation. They rightly state that this is not something we can count on. Why is this so important?

Modules give us more than just a way to specify local variables; they are not just procedures grown up from Pascal or ALGOL-60. Modules are a tool that you can use to preserve the values of variables and to hide information - about *how* some algorithm gets done, as well about data and structure.

We will start by looking at a program module called *ModuleTest.mod* (see Listing 1 at the end of this article) that will define a local module called *Widget*.

'Widget' as an identifier is a great teaching tool. It is totally sexless and doesn't offend anyone, except for a few anal-retentive types, and you will see it a lot. I use 'widget' as an identifier for all kinds of things. Don't get use to just one type associated with 'widget'; that is, don't assume that 'widget' will always be a local module or anything else.

What *ModuleTest.mod* will attempt to show is the ability of a local module to implement static variables.

When a PROCEDURE is called in Pascal or Modula-2, the variables declared in the PROCEDURE exist only during its execution. However the module's lifespan is longer than that of a normal PROCEDURE, and the variables declared within a module live as long as the module itself.

Consider our modules declared in the program module below. We will be watching the values of two variables: *TestVar1* and *TestVar2*.

Widget is a local module, and we don't have any idea when it will be initialized. We can be sure, however, that it will have been initialized before we execute *TestVars*. When the module is initialized, TestVar1 will be given whatever value is left after the execution of the FOR (in our case it will be 10), and *TestVar2* will have the ASCII codes of the characters "A".."J" as its elements.

So on the first call to *TestVars* from *ModuleTest*, we will see the following as our output:

```
TestVar2's value is : ABCDEFGHIJ
TestVar1's value is :        10
Adding 1 to TestVar1.
```

You should note that the values of the *TestVar1* and *TestVar2* variables are preserved from call to call of the routine *TestVars*. Their values don't vanish; they aren't recreated for a subsequent call.

How can we be sure the values aren't recreated from call to call? Okay, skeptic! The only other way our test results could be achieved is if the *Widget* initialization were done repeatedly each time *TestVars* was called. Add a semicolon to the second-last line in *Widget*'s main code ("END (* FOR *)"), then add the lines:

```
WriteString ( "We've been here." );
WriteLn
```

Now run the program again and see if you have more than one of the "We've been here." signs. Satisfied? Can we go on now? Okay.

What would have happened if we had declared the vars inside of the PROCEDURE *TestVars*, rather than at the module global level?

Look at the second program module - *ModuleTest2.mod* (listing 2). This time we have added some lines of code and two new declarations within the PROCEDURE *TestVars*.

The two new variables are *ProcVar1* and *ProcVar2*. These are declared solely within the *TestVars*, and only exist during the execution of the PROCEDURE.

When you execute this module, the first part of the output will look like this:

```
ProcVar2's value is : ABCDEFGHIJ
ProcVar1's value is :         0
TestVar2's value is : ABCDEFGHIJ
TestVar1's value is :        10
Adding 1 to TestVar1.
```

But the second part will look something like this:

```
ProcVar2's value is : <<
ProcVar1's value is :      2560
TestVar2's value is : ABCDEFGHIJ
TestVar1's value is :        11
Adding 1 to TestVar1.
```

Notice that the values of the *ProcVar*s are 'undefined'! - they didn't remain intact across repeated calls to *TestVars*.

EXPORT QUALIFIED and an old *Concat*

Let's leave static variables now and take a look at EXPORT QUALIFIED. This is used from within local modules to help you in distinguishing the local identifiers of local modules from those belonging to IMPORTed modules.

Prior to the 3rd edition of *Programming in MODULA-2*, Niklaus Wirth had defined EXPORT QUALIFIED as something you could include in a DEFINITION MODULE to force qualification of an identifier that was exported from a module. (Qualification refers to the explicit module name prefixed to an identifier using the dot conjunction. For example, if a module had the identifier *Widget* in a module called *SolarMagic*, the qualified form of the identifier would be *SolarMagic.Widget*.) Forcing qualification is sometimes a very handy tool. In the 3rd edition, however, Wirth seems to have changed the rules of Modula-2, and now has EXPORT QUALIFIED defined only as an element of local modules.

To illustrate the use of EXPORT QUALIFIED, try the module in Listing 3 using your Modula-2 and the *Strings* module included in your support library. (Follow the syntax diagrams on pages 198, 197 and 196 of *Programming in MODULA-2*.)

Compile the program and see what happens on your compiler. You may be surprised at the results.

There are three questions in the comments of this code:

1. Do you know why "IMPORT InOut;" must be imported entirely without the FROM qualifier?
2. Which *Concat* is executed - the *Concat* from module *Strings* or the one from the local module?
3. Why must the *WriteString* and *WriteLn* be qualified?

The answers to questions one and three are related. Exported items from *InOut* must be qualified because of the way they were IMPORTed (without the FROM qualification). The reason that we can't use the FROM qualifier is that the definition of the language requires us to import the entire module if we IMPORT any part of it into a local module.

As for question two, it is the *Concat* from *Strings* that gets executed. The logic of the compiler 'knows' that the local module's *Concat* must be qualified, since it is declared "EXPORT QUALIFIED". The *Concat* isn't explicitly qualified, so it is implicitly qualified as the *Strings* module's *Concat*. How do we know that the latter is the one executed? Because of the lack of a '*' in the final output.

Some of you with old TDI compilers, may be surprised at what kind of output you get. Let us look for a moment at the *Concat* supplied in the first couple of versions of the TDI *Strings* module. The comments in the definition of the *Concat* PROCEDURE indicate that:

```
Result = S1 + S2
```

This is exactly how you want *Concat* to work! However, the catch is in how the PROCEDURE within the module was actually implemented. In the newer versions of the compiler, the *Strings* module has been fixed; in the first few, however, when *Concat* first began execution it did an explicit assignment of *S1* to *Result*. If *S2* and *Result* were the same, the value in *S2* was destroyed. To see what *should* have happened, we might use the statement:

```
Concat ( Widget, Wodget, Wodget)
```

In string logic, this means: take the string *Widget*, place the string *Wodget* on the end of it, then place the entire string into *Wodget*. The original *Strings* module unfortunately copies *S1* into *Result*. This makes *S1* equal to *S2*, then copies *S2* into itself (which does nothing but use time). The upshot could be expressed as:

```
Result = S1 if ADR (Result) = ADR (S2)
```

This is just one of those little errors that programmers sometimes make. Clearly though, there was no code review or walk-through before these modules were released. The code shown here in the local module for *Concat* fixes the problem with the early *Strings* modules. The logic explicitly looks for:

```
ADR (Result) = ADR (S2)
```

If the two ADR's are equal, string *S2* is preceded with blanks, and *S1* is inserted into *S2*. So to see which of the *Concat*s is executed, try the following code in the main program:

```
Assign ( String1, "boo boo!" );
Concat ( "Oops ", String1, String1 );
InOut.WriteString ( String1 );
InOut.WriteLn;
```

Does it "Oops boo boo!"? Or does it just "Oops boo boo!*"? Or (if you are using the older modules, does it just "boo boo!"? Feel free to use this piece of code in your own module (perhaps called *MyStrings*).

We have now explained the answer to question two. Since the output of the module is "First", and not "First Second", its was the *Concat* from the *Strings* module was executed.

CASE Compilation OF

Executable code size can be deceiving. It can lead uninformed people to spout off about how compressed their code is because of the language they are using. A typical offender is the C programmer who claims that C produces smaller code than Pascal or "any other High Level Language"!

The size of code is a product of the compiler writer and, to a lesser degree, the programmer. Knowing how your compiler lexically analyizes, tokenizes, generates code, and compresses the generated code, is critical to picking the proper statements and structures to use when coding your programs.

In the listings at the end of this article are two versions of a program that is useful to folks with multiple printers. The problem with multiple printers is you never know which printer Preferences was last set to, and AmigaDOS doesn't have a simple way to show you. The Module *ShowPrinter* simply prints out what is in your Preferences file.

The two modules below, *ShowPrinterELSIF* (Listing 4) and *ShowPrinterCASE* (Listing 5) do exactly the same thing. The only difference between them is that one uses an ELSIF and the other uses CASE statements. The generated code for the two is, however, dramatically different. *ShowPrinterELSIF* has a size of 10976 bytes; *ShowPrinterCASE* has 16580 bytes. The difference is in the way the code is generated.

Let's look at CASE statments in particular. The CASE statement in Modula-2 has the following structure:

```
<case_statement> ==>
   "CASE" <case_expression> "OF" <case> { "|" <case>}
   [ELSE <statement_sequence>]
   "END".

<case> ==>
   [<case_label_list> ":" <statement_sequence>].

<case_label_list> ==>
   <case_label> {"," <case_label>}.

<case_label> ==>
   <const_expression> [".." <const_expression>].
```

What a programmer needs to look at is the compiler's ability to compress the range of the <case_label_list>.

For those of you who don't read BNF, a short note is in order so that you may also participate. The rest of you can skip over the following paragraph.

BNF is a method for representing a grammer symbolically. The initials stand for 'Backus-Naur Form'. As an example, consider the following CASE statement:

```
CASE Widget OF
   Wodget :  FirstStatmentOfWodget;
             SecondStatementOfWodget |
   Wadget :  FirstStatementOfWadget;
             SecondStatementOfWadget
   ELSE
             FirstStatementOfElse;
             SecondStatementOfElse
END (* CASE *);
```

Widget is representative of where you find **case_expression**s. Wodget and Wadget are **case_label**s. Now you may continue.

The problem a programmer is likely to run into is two fold. First, what kind of code does a CASE statement generate when compared to an ELSIF statement? Secondly, what kind of code does the CASE statement generate itself?

The problem that the CASE statement presents to the compiler writer is the handling of the range of **case_label** values.

CASE statements are usually generated as jump tables by a compiler, whereas ELSIF statements are usually generated as separate tests and branches.

In the CASE, statement the compiler writer has to decide how to build the jump table. If the base type of the case expression is BYTE, then 256 different jumps are possible. The programmer doesn't normally use all 256 jumps, however; in fact, he or she is likely to use only a small portion of the actual ordinal range of the base type of the case expression (that is, the number of actual **case_label**s tends to be small compared to how many there *could* be). Most compilers therefore simply look at the maximum and minimum values of the **case_label**s actually used, and generate jump tables of the minimum size that will accommodate the range. For example, if your smallest **case_label** value is 3, and your largest is 10, you would expect to get a jump table of eight entries.

Some compilers, however, take the point of view that, if the base type is CARDINAL, you ought to pay the price of not setting up a sub-range for what you need, and they do some nasty things without telling you, like allocating 32767 jumps (of 32 bits each). Even worse, though far more consistent, you could get 65535 jump locations.

To get smaller code, you should develop a new type or sub-range representing the range for that CASE. This usually means that you have to clutter up your code to create something that is just not as meaningful as the original CASE.

As the old adage goes, you can write confusing code in any language.

In the examples of *ShowPrinterELSIF* and *ShowPrinterCASE*, the two programs generate dramatically different code.

Using one compiler, the size of the ELSIF code comes out to 1,410 bytes, while the CASE code is 3,753 bytes - a 2,343 byte difference. Why? There are four routines in the program whose code has been converted from ELSIF to CASE statements between the two modules. The compiler generates 512 bytes for each CASE statement that doesn't have an implied (via base type) shorter **case_label** range. Multiplying 4 by 512 gives 2,048, which leaves just 295 bytes to account for. That amount is the difference (in *this* compiler) between the code generated for the testing ELSIF statements and that for the jump table CASE statements.

The tradeoff here is the ever familiar one between memory and speed: more memory for the faster executing CASE, or less memory for the (sometimes) slower to execute ELSIF. (Actually the ELSIF is slower if a loop is repeated or if the test must go beyond the first conditional.)

Not all compilers work in the same way as the one I used for this example. Some do a good job of **case_label** range compression. You as a programmer should investigate your compiler and look for which of these tradeoffs you should use.

You must then decide when to exercise your tradeoff decision in favour of the ELSIF rather than the CASE. That is one of the things that 'designing' your code is all about.

Listing 1: ModuleTest

```
MODULE ModuleTest;

(*
   If you import some portion of a module from a local
   module you must import the entire module from the
   global module.  This is straight out of Wirth's
   definition and not a problem with TDI's Modula-2.
*)
IMPORT InOut;

VAR (* Global vars for the MODULE ModuleTest *)
 (* A utility variable used in a FOR loop. *)
 LoopCounter : CARDINAL;

(*
   Declaration of the local Module goes here.
*)
MODULE Widget;
(*
   Local modules look exactly like program modules except
   that they end with an END; not an END.
*)

(*
   Note that the procedures below need not be qualified
   in the code and that we can include this import only
   because of the IMPORT InOut statement in global module.
*)
FROM InOut IMPORT WriteCard, WriteLn, WriteString;

(*
   We will use EXPORT TestVars in this example but in the
   next example we will use the QUALIFIED modifier to show
   the differences.
*)
EXPORT (* the procedure we will use for testing *) TestVars;

TYPE
 CharArray10 = ARRAY [ 0..9 ] OF CHAR;

(*
   These are global to Widget but they are Static Vars.
*)
VAR
 TestVar1 : CARDINAL;
 TestVar2 : CharArray10;

PROCEDURE TestVars;

BEGIN (* Test Vars *)
 WriteString ( "TestVar2's value is : " );
 WriteString ( TestVar2 );
 WriteLn;

 WriteString ( "TestVar1's value is : " );
 WriteCard   ( TestVar1, 8 );
 WriteLn;

 WriteString ( "Adding 1 to TestVar1." );
 WriteLn;
 TestVar1 := TestVar1 + 1
END TestVars;

BEGIN (* Widget *)
 FOR TestVar1 := 0 TO 9 DO
  TestVar2 [ TestVar1 ] := CHR ( TestVar1 + ORD ( "A" ) )
 END (* FOR *)
END Widget;

BEGIN (* Module Test *)
 FOR LoopCounter := 1 TO 10  DO
  TestVars;
  InOut.WriteLn
 END (* FOR *)
END ModuleTest.
```

Listing 2: ModuleTest2

```
MODULE ModuleTest2;

(*
   If you import some portion of a module from a local
   module you must import the entire module from the
   global module.  This is straight out of Wirth's
   definition and not a problem with TDI's Modula-2.
*)
IMPORT InOut;

VAR (* Global vars for the MODULE ModuleTest *)
 (* A utility variable used in a FOR loop. *)
 LoopCounter : CARDINAL;

(*
   Declaration of the local Module goes here.
*)
MODULE Widget;
(*
   Local modules look exactly like program modules except
   that they end with and END; not an END.
*)

(*
   Note that the procedures below need not be qualified
   in the code and that we can include this import only
   because of the IMPORT InOut statement in global module.
*)
FROM InOut IMPORT WriteCard, WriteLn, WriteString;

(*
   We will use EXPORT TestVars in this example but in the
   next example we will use the QUALIFIED modifier to show
   the differences.
*)
EXPORT (* the procedure we will use for testing *) TestVars;

TYPE
 CharArray10 = ARRAY [ 0..9 ] OF CHAR;

(*
   These are global to Widget but they are Static Vars.
*)
VAR
 TestVar1 : CARDINAL;
 TestVar2 : CharArray10;

PROCEDURE TestVars;

VAR
 ProcVar1 : CARDINAL;
 ProcVar2 : CharArray10;

BEGIN (* Test Vars *)
(*
   Initialize the PROCEDURE variables to some first
   time value.  Make sure it is only done once.
*)
 IF TestVar1 = 10
  THEN
   ProcVar1 := 0;
   ProcVar2 := TestVar2
 END;

 WriteString ( "ProcVar2's value is : " );
 WriteString ( ProcVar2 );
 WriteLn;

 WriteString ( "ProcVar1's value is : );
 WriteCard   ( ProcVar1, 8 );
 WriteLn;

 WriteString ( "TestVar2's value is : " );
 WriteString ( TestVar2 );
 WriteLn;

 WriteString ( "TestVar1's value is : " );
 WriteCard   ( TestVar1, 8 );
 WriteLn;

 WriteString ( "Adding 1 to TestVar1." );
 WriteLn;
 TestVar1 := TestVar1 + 1
END TestVars;

BEGIN (* Widget *)
 FOR TestVar1 := 0 TO 9 DO
  TestVar2 [ TestVar1 ] := CHR ( TestVar1 + ORD ( "A" ) )
 END (* FOR *)
END Widget;

BEGIN (* Module Test *)
 FOR LoopCounter := 1 TO 10  DO
  TestVars;
  InOut.WriteLn
 END (* FOR *)
END ModuleTest2.
```

Listing 3: ShowDifferenceInExportQualified

```
MODULE ShowDifferenceInExportQualified;

(* Do you know why this has to be IMPORTed this way? *)
IMPORT InOut;

FROM Strings IMPORT Concat, Assign, Length;

VAR
 String1 : ARRAY [ 0..70 ] OF CHAR;
MODULE MakeNewStuff;
 FROM InOut IMPORT WriteString; WriteLn;
 EXPORT QUALIFIED Concat;

 PROCEDURE Insert (     SourceString  : ARRAY OF CHAR;
        VAR TargetString  : ARRAY OF CHAR;
          StartPosition : CARDINAL       );
 VAR
  NextPosition : CARDINAL;
  SourceLength : CARDINAL;

 BEGIN (* Insert *)
  SourceLength := Length ( SourceString );
  IF     ( SourceLength > 0 )
     AND ( StartPosition <= HIGH ( TargetString ) )
   THEN (* there is something to do. *)
    IF SourceLength + StartPosition > HIGH ( TargetString )
     THEN (* We only need to copy to the HIGH point. *)
      StopPosition := HIGH ( TargetString )
     ELSE (* We need to do all the source string chars. *)
      StopPosition := SourceLength + StartPosition
    END (* of IF *);
    NextPosition := StartPosition;
    WHILE ( NextPosition <= StopPosition ) DO
     TargetString [ NextPosition ] :=
      SourceString [ NextPosition - StartPosition ];
     NextPosition := NextPosition + 1
    END (* of WHILE *)
  END (* of IF *)
 END Insert;

 PROCEDURE Concat (             VAR S1,
                  (* With   *)     S2     : ARRAY OF CHAR;
                  (* giving *) VAR Result : ARRAY OF CHAR);

   (* equivalent to   Result := S1 + S2    *)

 VAR
  LenS1           : CARDINAL;
  LenS2           : CARDINAL;
  Index           : CARDINAL;

  AddressOfS1     : ADDRESS;
  AddressOfS2     : ADDRESS;
  AddressOfResult : ADDRESS;
```

```
 BEGIN (* Concat *)
   AddressOfS1 := ADR ( S1 );
   AddressOfS2 := ADR ( S2 );
   AddressOfResult := ADR ( Result );

   LenS1 := Length(S1);
   LenS2 := Length(S2);

   IF (LenS1 + LenS2) = 0
    THEN Result [ 0 ] := Terminator
    ELSIF (LenS1 + LenS2) < ( HIGH ( Result ) + 2 )
     THEN (* There is something to do so.... *)
      IF AddressOfS2 = AddressOfResult
       THEN (* S2 & Result are the same so, Move as many
               characters as there are in S2 to the right
               by amount of characters there are in S1.   *)

        FOR I := LenS2 - 1 TO 0 DO
         Result [ LenS1 + I ] := S2 [ I ]
        END (* of FOR *);

        (* Now stick in the characters of S1 *)
        Insert ( S1, Result, 0 )
       ELSE (* Stick S2 on end of S1 *)
        (* Check to see if move is necessary *)
        IF AddressOfS1 <> AddressOfResult
         THEN (* moving characters is necessary *)
          Assign ( Result, S1 )
        END (* of IF *);

        FOR I := LenS1 TO ( LenS1 + LenS2 - 1 ) DO
         Result [ I ] := S2 [ I - LenS1 ]
        END (* FOR *)
      END (* of IF *);

(* Special code to indicate which result we are using.*)
      Result [ LenS1 + LenS2 ] := "*";

      Result [ LenS1 + LenS2 + 1 ] := Terminator
   END (* IF *)
 END Concat;

BEGIN (* MakeNewStuff *)
END MakeNewStuff;

BEGIN (* Show Difference In Export Qualified *)
 Assign ( String1, " Second." ) (* Initialize String1. *);

 Concat ( "First", String1, String1 )
 (* but which Concat is this?  The one from Strings
    or the one from the local module? *);

 (* Do you know why these lines of
    code have to be qualified? *)
 InOut.WriteString ( String1 );
 InOut.WriteLn
END ShowDifferenceInExportQualified.
```

Listing 4: ShowPrinterELSIF

```
MODULE ShowPrinterELSIF;
(*
  (c) COPYRIGHT Software Industry & General Hardware 1988
  (c) COPYRIGHT Software Industry & General Hardware 1987
  (c) COPYRIGHT Software Industry & General Hardware 1986
*************************************************************
*                                                           *
* This program and its co-routines may be used freely and   *
* distributed freely with the permission of S.I.G.H.   The *
* program does not lie in the realm of Public Domain.  All *
* rights are reserved to S.I.G.H.  Permission may be        *
* obtained by writing to S.I.G.H. or by phone.              *
*          Software Industry & General Hardware             *
*          POB 17201                                        *
*          Irvine, CA 92713 USA                             *
*          714.847.7823                                     *
*                                                           *
*************************************************************
*)
(* Use BSR's instead of JSR's to procedures in module. *)
(*$Q+*)
(* Turn off range checking. *)
(*$R-*)
(* Turn off stack checking. *)
(*$S-*)
(* Turn off overflow checking. *)
(*$V-*)

IMPORT Trapper;

FROM SYSTEM      IMPORT ADR,           ADDRESS,
                        BYTE,          TSIZE;
FROM Intuition   IMPORT IntuitionBase, IntuitionName;
FROM ASCII       IMPORT NUL;
FROM Libraries   IMPORT OpenLibrary,   CloseLibrary,
                        LibraryPtr;
FROM Preferences IMPORT Preferences,   GetPrefs,
                        Elite,         Fine,
                        Draft,         Letter,
                        SixLPI,        EightLPI,
                        Baud110,       Baud300,
                        Baud1200,      Baud2400,
                        Baud4800,      Baud9600,
                        Baud19200,     BaudMIDI,
                        SerialPrinter, ParallelPrinter,
                        Pica                          ;
FROM InOut       IMPORT WriteString,   WriteLn,
                        WriteCard;

CONST
 NULL       = ADDRESS ( 0 );

VAR
 PrefBuffer : Preferences;
 Widget     : ADDRESS;

PROCEDURE ShowPort;

BEGIN (* Show Port *)
 WriteString ( "the " );
 IF CARDINAL ( PrefBuffer.PrinterPort ) = ParallelPrinter
  THEN WriteString ( "parallel" )
  ELSE WriteString ( "serial" )
 END;

 WriteString ( " port in use" );

 IF CARDINAL ( PrefBuffer.PrinterPort ) = SerialPrinter
  THEN
   WriteString ( " running at " );
   IF CARDINAL ( PrefBuffer.BaudRate ) = Baud110
    THEN WriteString ( "110"  )
   ELSIF CARDINAL ( PrefBuffer.BaudRate ) = Baud300
    THEN WriteString ( "3"    )
   ELSIF CARDINAL ( PrefBuffer.BaudRate ) = Baud1200
    THEN WriteString ( "12"   )
   ELSIF CARDINAL ( PrefBuffer.BaudRate ) = Baud2400
    THEN WriteString ( "24"   )
   ELSIF CARDINAL ( PrefBuffer.BaudRate ) = Baud4800
    THEN WriteString ( "48"   )
   ELSIF CARDINAL ( PrefBuffer.BaudRate ) = Baud9600
    THEN WriteString ( "96"   )
   ELSIF CARDINAL ( PrefBuffer.BaudRate ) = Baud19200
    THEN WriteString ( "192"  )
   ELSE WriteString ( "MIDI" )
   END (* of IF *);

   IF     ( CARDINAL ( PrefBuffer.BaudRate ) > Baud110  )
      AND ( CARDINAL ( PrefBuffer.BaudRate ) < BaudMIDI )
    THEN WriteString ( "00" )
   END (* of IF *);
   WriteString ( " baud" )
 END (* of IF *);
 WriteString ( "." ); WriteLn
END ShowPort;
```

```
PROCEDURE ShowPitch;

 BEGIN
  IF PrefBuffer.PrintPitch = Pica
   THEN WriteString ( "Pica"  )
  ELSIF PrefBuffer.PrintPitch = Elite
   THEN WriteString ( "Elite" )
  ELSIF PrefBuffer.PrintPitch = Fine
   THEN WriteString ( "Fine"  )
  ELSE WriteString ( "Unknown"  )
  END (* of IF *);
  WriteString ( " pitch." )
 END ShowPitch;

PROCEDURE ShowQuality;

BEGIN (* Show Quality *)
 IF PrefBuffer.PrintQuality= Draft
  THEN WriteString ( "Draft"   )
 ELSIF PrefBuffer.PrintQuality = Letter
  THEN WriteString ( "Letter"  )
 ELSE WriteString ( "Unknown" )
 END (* of IF *);
 WriteString ( " quality print." )
END ShowQuality;

PROCEDURE ShowSpacing;

BEGIN
 IF PrefBuffer.PrintSpacing = SixLPI
  THEN WriteString ( "Six" )
  ELSIF PrefBuffer.PrintSpacing = EightLPI
   THEN WriteString ( "Eight" )
  ELSE WriteString ( "Unknown" )
 END (* of IF *);
 WriteString ( " lines per inch." )
END ShowSpacing;

PROCEDURE ShowMargin;

BEGIN
 WriteString ( "Width " );
 WriteCard   (   PrefBuffer.PrintRightMargin
               - PrefBuffer.PrintLeftMargin
               + 1                          , 8 );
 WriteString ( " characters per line." ); WriteLn;

 WriteString ( "      " );
 WriteCard   ( PrefBuffer.PrintLeftMargin, 8 );
 WriteString ( " (left margin)" ); WriteLn;

 WriteString ( "      " );
 WriteCard   ( PrefBuffer.PrintRightMargin, 8 );
 WriteString ( " (right margin)" ); WriteLn
END ShowMargin;

PROCEDURE ShowPrinterPreferences;

TYPE
 PrinterPreferenceKind = ( Port,    Pitch, Quality,
                           Spacing, Margin         );

VAR
 Index : PrinterPreferenceKind;

BEGIN
 FOR Index := Port TO Margin DO
  CASE Index OF
   Port    : ShowPort    |
   Pitch   : ShowPitch   |
   Quality : ShowQuality |
   Spacing : ShowSpacing |
   Margin  : ShowMargin  |
   ELSE (* do nothing *)
  END (* of CASE *);
  WriteLn
 END (* FOR *)
END ShowPrinterPreferences;

BEGIN (* Show Printer *)
 IF ( IntuitionBase = NULL )
  THEN IntuitionBase := OpenLibrary ( IntuitionName, 0 )
 END;

 IF ( IntuitionBase <> NULL )
  THEN
   Widget := GetPrefs ( PrefBuffer, TSIZE ( Preferences ) );

   WriteString
    ( "Your currently preferred printer is the " );
   IF PrefBuffer.PrinterFilename [ 0 ] <> NUL
    THEN
     WriteString ( PrefBuffer.PrinterFilename );
     WriteString ( " with:" ); WriteLn;
     ShowPrinterPreferences
    ELSE WriteString ( " <NONE> " )
   END (* of IF *)
 END (* of IF *);

 IF IntuitionBase <> NULL
  THEN CloseLibrary ( IntuitionBase )
 END (* of IF *)
END ShowPrinter.
```

Listing 5: ShowPrinterCASE

```
MODULE ShowPrinterCASE;
(*
  (c) COPYRIGHT Software Industyr & General Hardware 1988
  (c) COPYRIGHT Software Industry & General Hardware 1987
  (c) COPYRIGHT Software Industry & General Hardware 1986

*************************************************************
*                                                           *
* This program and its co-routines may be used freely and   *
* distributed freely with the permission of S.I.G.H. as     *
* long as this header is included in the distribution of    *
* the source.                                               *
*                                                           *
*          Software Industry & General Hardware             *
*          713 Hunter Street                                *
*          Ramona, CA 92065 USA                             *
*                                                           *
* This Program was developed using Modula-2/Amiga           *
* available from TDI Software, Inc., Dallas, Texas.         *
*                                                           *
*************************************************************
*)

(* Use BSR's instead of JSR's to procedures in module. *)
(*$Q+*)

(* Turn off range checking. *)
(*$R-*)

(* Turn off stack checking. *)
(*$S-*)

(* Turn off overflow checking. *)
(*$V-*)

IMPORT Trapper;

FROM SYSTEM      IMPORT ADR,           ADDRESS,
                        BYTE,          TSIZE;
FROM Intuition   IMPORT IntuitionBase, IntuitionName;
FROM ASCII       IMPORT NUL;
FROM Libraries   IMPORT OpenLibrary,   CloseLibrary,
                        LibraryPtr;
```

```
FROM Preferences IMPORT Preferences,   GetPrefs,
                        Pica,          Elite,
                        Fine,          Draft,
                        Letter,        SixLPI,
                        EightLPI,      Baud110,
                        Baud300,       Baud1200,
                        Baud2400,      Baud4800,
                        Baud9600,      Baud19200,
                        BaudMIDI,      SerialPrinter,
                        ParallelPrinter            ;
FROM InOut       IMPORT WriteString,   WriteLn,
                        WriteCard;

CONST
 NULL       = ADDRESS ( 0 );

VAR
 PrefBuffer : Preferences;
 Widget     : ADDRESS;

PROCEDURE ShowPort;

BEGIN (* Show Port *)
 WriteString ( "the " );
 IF CARDINAL ( PrefBuffer.PrinterPort ) = ParallelPrinter
  THEN WriteString ( "parallel" )
  ELSE WriteString ( "serial" )
 END;

 WriteString ( " port in use" );

 IF CARDINAL ( PrefBuffer.PrinterPort ) = SerialPrinter
  THEN
   WriteString ( " running at " );
   CASE CARDINAL ( PrefBuffer.BaudRate ) OF
    Baud110   : WriteString ( "110"  ) |
    Baud300   : WriteString ( "3"    ) |
    Baud1200  : WriteString ( "12"   ) |
    Baud2400  : WriteString ( "24"   ) |
    Baud4800  : WriteString ( "48"   ) |
    Baud9600  : WriteString ( "96"   ) |
    Baud19200 : WriteString ( "192"  ) |
    ELSE WriteString ( "MIDI" )
   END (* of IF *);

   IF     ( CARDINAL ( PrefBuffer.BaudRate ) > Baud110  )
      AND ( CARDINAL ( PrefBuffer.BaudRate ) < BaudMIDI )
    THEN WriteString ( "00" )
   END (* of IF *);
   WriteString ( " baud" )
 END (* of IF *);
 WriteString ( "." ); WriteLn
END ShowPort;
PROCEDURE ShowPitch;

 BEGIN (* Show Pitch *)
  CASE PrefBuffer.PrintPitch OF
   Pica  : WriteString ( "Pica"  ) |
   Elite : WriteString ( "Elite" ) |
   Fine  : WriteString ( "Fine"  ) |
   ELSE WriteString ( "Unknown"  )
  END (* of CASE *);
  WriteString ( " pitch." )
 END ShowPitch;

PROCEDURE ShowQuality;

BEGIN (* Show Quality *)
 CASE PrefBuffer.PrintQuality OF
  Draft  : WriteString ( "Draft"   ) |
  Letter : WriteString ( "Letter"  ) |
  ELSE     WriteString ( "Unknown" )
 END (* of CASE *);
 WriteString ( " quality print." )
END ShowQuality;

PROCEDURE ShowSpacing;

BEGIN (* Show Spacing *)
 CASE PrefBuffer.PrintSpacing OF
  SixLPI   : WriteString ( "Six" )   |
  EightLPI : WriteString ( "Eight" ) |
  ELSE WriteString ( "Unknown" )
 END (* of CASE *);
 WriteString ( " lines per inch." )
END ShowSpacing;

PROCEDURE ShowMargin;

BEGIN (* Show Margin *)
 WriteString ( "Width " );
 WriteCard   (   PrefBuffer.PrintRightMargin
               - PrefBuffer.PrintLeftMargin
               + 1                          , 8 );
 WriteString ( " characters per line." ); WriteLn;

 WriteString ( "      " );
 WriteCard   ( PrefBuffer.PrintLeftMargin, 8 );
 WriteString ( " (left margin)" ); WriteLn;

 WriteString ( "      " );
 WriteCard   ( PrefBuffer.PrintRightMargin, 8 );
 WriteString ( " (right margin)" ); WriteLn
END ShowMargin;
PROCEDURE ShowPrinterPreferences;

TYPE
 PrinterPreferenceKind = ( Port,    Pitch, Quality,
                           Spacing, Margin          );

VAR
 Index : PrinterPreferenceKind;

BEGIN (* Show Printer Preferences *)
 FOR Index := Port TO Margin DO
  CASE Index OF
   Port    : ShowPort    |
   Pitch   : ShowPitch   |
   Quality : ShowQuality |
   Spacing : ShowSpacing |
   Margin  : ShowMargin  |
   ELSE (* do nothing *)
  END (* of CASE *);
  WriteLn
 END (* FOR *)
END ShowPrinterPreferences;

BEGIN (* Show Printer *)
 IF ( IntuitionBase = NULL )
  THEN IntuitionBase := OpenLibrary ( IntuitionName, 0 )
 END (* of IF *);

 IF ( IntuitionBase <> NULL )
  THEN
   Widget := GetPrefs ( PrefBuffer, TSIZE ( Preferences ) );

   WriteString
    ( "Your currently prefered printer is the " );

   IF PrefBuffer.PrinterFilename [ 0 ] <> NUL
    THEN
     WriteString ( PrefBuffer.PrinterFilename );
     WriteString ( " with:" ); WriteLn;
     ShowPrinterPreferences
    ELSE WriteString ( " <NONE> " )
   END (* of IF *)
 END (* of IF *);

 IF IntuitionBase <> NULL
  THEN CloseLibrary ( IntuitionBase )
 END (* of IF *)
END ShowPrinter.
```

Programming in M2AMIGA

An integrated Modula-2 development system

by Anthony Bryant

Anthony Bryant is a computer engineer working with remote control systems and robotics. He programs the Amiga and other Commodore computers in his spare time.

The M2AMIGA Software Development System works with v1.2 of the Amiga OS and runs on the Amiga 500, 1000 and 2000 with at least one disk drive and at least 512 kilobytes of RAM. Booting with the M2AMIGA disk brings up the Workbench and a CLI window. The system can be used either way, providing icons if desired. The basic system multitasking components are *m2c* (the compiler), *m2l* (the linker) and *m2emacs* (the editor).

In the Workbench environment, source files edited in *m2emacs* (the default source tool) automatically create source icons. Shift-clicking on a source icon and *m2c* runs the compiler and automatically creates the object icon. If there are errors in compilation, then double-clicking on the source icon brings up *m2emacs* and reads the compiler's error output, placing the cursor at the next error for fixing. If there are no errors, then double-clicking on the object icon runs *m2l* (the default object tool) and outputs the executable.

In the CLI environment, you invoke each system component in turn. No icons are created unless you want them. A supplied program, *m2error*, reads error messages without using the editor. It's much easier, though, to fix errors as you read them, using the editor.

One of the best features of this package (and one that contributes to making this a complete development system designed especially for the Amiga) is the organization of project directories. This feature is managed with a program called *m2project*, which creates your project directory and three subdirectories - *sym*, *obj* and *ref* - for storing compiler generated files. A user-configurable script file called *m2path* contains local and global path information that all system components can examine to find needed files. Simple and very effective!

Another noteworthy feature of this package is the inclusion of "pre-compiled" library modules, which make a big contribution to the speed of compilation. M2AMIGA supplies the basic Amiga Libraries (Dos, Exec, Intuition, etc.), the standard M2 libraries (InOut, FileSystem, Heap, etc.) and some special M2AMIGA libraries, such as FFP and other math conversion libraries, which are Amiga-specific.

The compiler

A compiler is always subject to some limitations, whether arising from the implementation of the language or the language itself. M2 is a language rich in command structures, but with the capability of being tailored to a particular CPU. System-specific extensions will of course vary, which is why they are deliberately separated using the FROM SYSTEM IMPORT line. BYTE, WORD, ADDRESS, BITSET, LONGSET and FFP are especially dependent on the Motorola 68000. Watch out for these when trying to port source!

Though a relatively new language, M2 is not without its older definitions - overtaken by time and events. For example, FLOAT has been replaced by REAL, LONGREAL and FFP to accommodate the IEEE 32-bit format, IEEE 64 bit format and Motorola Fast Floating Point Format, respectively. It should be mentioned that FFP and LONGREAL arithmetic are performed using the Amiga's libraries (in ROM and on disk) but type REAL is an M2 software emulation. Their use in a program automatically loads the required math library.

When working at the system level, it is often necessary to give a variable a new type. Recent improvements in the C language standard attest to the importance now given to types in that

language. In M2AMIGA, types are normally aligned to byte boundaries, except types requiring a word or a long word, which always start at word boundaries (as required by the 68000 CPU). In a type transfer, the size of the original type must be identical to that of the destination type. The SYSTEM-specific CAST procedure carries out a type transfer. Thus, undefined type BYTE can only be CAST as type CHAR. But it is not always clear, especially when porting, whether the source and destination types both occupy the same amount of memory. For this reason a type conversion is recommended, rather than a type transfer.

```
BITSET(cardinal)
     converts a CARDINAL variable to type BITSET
INTEGER(float)
     converts a REAL or FFP variable to type INTEGER
FFP(integer)
     converts an INTEGER variable to type FFP
```

This makes procedures such as ORD(), CHR(), FLOAT() and VAL() unnecessary, but they are included for the sake of backward compatibility.

```
     INTEGER("A")  corresponds to ORD("A")
     CHAR(65)      corresponds to CHR(65)
     REAL(100)     corresponds to FLOAT(100)
```

The SIZE() procedure yields the size in bytes of practically everything: INTEGERs, ARRAYs, RECORDs, etc., obviating the need for the older TSIZE, which is included anyway.

M2AMIGA's SHIFT() procedure shifts the bit pattern of a given value left or right *x* places, provided the shift is possible (convert an INTEGER to a LONGINT if you want to shift a variable more than 16 bits, for example). Some M2 implementations provide for arithmetic and logical shifts, as well as arithmetic rotate and two's complement, but these are not standard and are not included in M2AMIGA. Is that turning a compiler into an assembler? Well, I haven't found a need for those (yet)! What I do miss, sometimes, is a bitwise AND and a bitwise OR - the brute force method (rather than finessing with SETs). At the end of this article I have included a little set of functions (*bitwiseOps.mod*) to aid in getting used to using SETs!

The INLINE() procedure inserts hex machine code into a program. This is great for those small jobs; one could even use mnemonic constants, like:

```
CONST rts = 4E75H; beq = 6700H; dbeq = 57C8H; lea = 41C0H;
```

There are only about 250 variant mnemonics to cope with! Mercy! INLINE() also allows you to store static numeric data in structures and arrays. A PROCEDURE normally starts with a chunk of code that fetches the module pointer, allocates memory for local variables, and passes parameters on the stack; it normally ends with a chunk of code that clears the stack, etc. The compiler can be instructed not to generate these two code chunks for specific procedures, for use with INLINE(). Since each element must be a word-sized (16 bits), however, this method is of limited value when you are trying to fill complex structures like a NewScreen or a NewWindow. It *does* work, but it is a very awkward way of doing what is straightforward in C - "pre-initializing" static structures in line with the code. M2AMIGA also provides no way to direct static data to CHIP or FAST RAM. An Amiga-specific command (borrowed from an assembler) like SECTION, CHIP, DATA would be useful here.

The compiler's task is to translate source text into machine code. The process is complex: symbols must be recognized, the syntax and data typing must be checked, internal tables must generated for structures encountered, and symbol (*sym*) files, object (*obj*) files and optional reference (*ref*) files must be output. A one-pass compiler like *m2c* carries out these operations in parallel, making it extremely fast, without any constraints on the programmer. The only stipulation is that all referenced elements must be declared before use.

The linker

For most stand-alone program development, *m2l*, the linker, is very fast, dependable, and all that one needs. As with all other M2AMIGA components designed especially for (rather than ported to) the Amiga, the linker is also multitasking and capable of searching current and library directories for needed *sym* and *obj* files, greatly speeding up this job.

But there are no options. The ability to link in an object file (with just pure data, say) generated by another language (such as an Assembler), or to link/load video or audio data into CHIP RAM, is just not available. M2AMIGA does not use the standard Amiga binary format for its object files.

Some special tools are on the way, though. Still under development are an object file converter, *m2oc*, which should solve the above programming problem; and *m2ll*, a library and device linker that will aid in developing new disk-based libraries. The completion of these additional tools should make the M2AMIGA Development System increasingly flexible.

The debug loader

The newest tool to be added to the M2AMIGA Development System is a source level debugger. It's often been said that once you have successfully compiled and linked your M2 program, you can be sure that it will run. Yes... and it will produce run-time errors!

Currently, with M2AMIGA, when run-time errors occur during testing, the program is interrupted with an AutoRequester that reports the cause of the error and permits the program to "abort", cleaning up after itself, but leaving the programmer none the wiser. Up until now, debugging consisted of a bunch of strategically-placed print statements, or carefully-written

BreakPoint() procedures, or even your own crude debug program linked in to Arts module's DebugProcedure().

Now, however, M2AMIGA includes *m2rd*, a linker/loader that, during the testing phase, replaces *m2l*. *m2rd* links all object files and loads and runs your program dynamically. When runtime errors occur (or a deliberate BreakPoint() or HALT command is executed), the familiar AutoRequester allows a second choice - "debug", which invokes the debugger.

The source level debugger

The debugger, *m2d*, comes up on a custom screen with five user-resizable windows.

Source text is displayed in the top window, with the current error position highlighted. This window has a scroll bar with which to view the entire text.

The second window, just below the source, displays the procedure call chain, with the most recently executed procedure at the top of the list. This window permits scrolling through the list of procedures called. The mouse is used for two actions here. Hitting the right button on one of the procedure entries puts up the source text for that procedure in the source window. Hitting the left button displays that procedure entry's local variables in a third window, called *Data1*. For simple variables, only the name, value and type are displayed. For structured types, extended zooming (also using the mouse) displays all substructures and contents. If you have pointer variables, you can also dereference these forward or backward. This intuitive approach allows you to browse through all the data available at the time of program termination.

The fourth window lists all of the modules currently loaded, giving the address of the global data block for normal modules, the library pointer for library modules, or all zeroes (NIL) for modules without implementation. As with the procedure window, the mouse buttons can be used to display source text for these modules in the source window at the top, or globally declared variables of the module in another data window, called *Data2*. Using the main menu, you can direct the two data windows, *Data1* and *Data2* independently to display procedure data or module data, as you like.

Basically, then, the debugger is linked in to the run-time system (Arts module), and takes a snapshot of your program at the time of the crash/interrupt. From inside your program, you can examine all modules, procedures and variables for which *obj* files, *ref* files and source text exist. Like all other M2AMIGA components, the disk is searched for every file needed according to *m2path*, and it can be used from either the Workbench or CLI.

Working with a prerelease version of the debugger, I was immediately struck by the ease of use and clear display of information. When the error in your source has been found (using the debugger) and fixed (using *m2emacs*), and the new *obj* and *ref* files have been generated (using *m2c*), you can, from within *m2rd*, unload the old module and run your program again, automatically loading only the new module. You see, all modules linked into the loader (*m2rd*) and everything ever loaded is kept in memory. If you restart a program again it starts immediately. In the interactive mode, you can unload a module, list all loaded modules, or clear a module's global data.

Clearly, then, if you want this *fast* debugging cycle, (with everything ever loaded kept in memory) you will need expanded memory, at least another 512k for programs IMPORTing many modules. Like the other multitasking components in M2AMIGA, though, *m2rd* will run on a stock 512k Amiga, allowing you to find the error's location and exit. I could quickly get used to life in this new era of source level debugging!

The future

The M2AMIGA Development System is continually being improved. There is version 3.2 of the *m2emacs* editor. In preparation for the next release, the compiler and linker have also been improved, and bugs in the initial release (3.1) of the Amiga library modules fixed. The standard M2 library modules have been augmented (Arts and Terminal), and expanded (Windows, Menus and Gadgets). All these are very interesting developments which should make M2 even easier to use!

Listing: bitwiseOps.mod

```
(* Bitwise Operators - BAND(), BOR(), BXOR(), BNOT()      *)

(* BITSET = SET OF [0..15]  i.e a 16-bit word, or
 * LONGSET= SET OF [0..31]  i.e a 32-bit word
 *)

(* function = n1 BAND n2    i.e. bitwise and all bits
 *)
PROCEDURE BAND(n1,n2: BITSET): BITSET;
BEGIN
   RETURN n1 * n2;  (* set intersection *)
END BAND;

(* function = n1 BOR n2     i.e. bitwise or all bits
 *)
PROCEDURE BOR(n1,n2: BITSET): BITSET;
BEGIN
   RETURN n1 + n2;  (* set union *)
END BOR;

(* function = n1 BXOR n2    i.e. bitwise xor all bits
 *)
PROCEDURE BXOR(n1,n2: BITSET): BITSET;
BEGIN
   RETURN n1 / n2;  (* symmetric set difference *)
END BXOR;

(* function = BNOT n1       i.e. complement all bits
 *)
PROCEDURE BNOT(n1: BITSET): BITSET;
BEGIN
   RETURN {0..15} - n1;  (* set difference *)
END BNOT;
```

Why I'd rather switch than fight

Switching over from "the real world"

by Monica Dobrov

Monica Dobrov is creative director at Beyond Graphics, and created the bit map of the Transactor logo used on the front cover. She's also a musician in the group "Death and Taxes".

If you happen to be one of the many individuals who have always used an Amiga and don't realize the day to day horrors of print production without one, perhaps you are taking things for granted. Why is desktop publishing so exciting? Please join me now on a little journey into the real world.

A typical job was an ad for a new restaurant: I received hastily scrawled copy, a menu, and a brief note reading "design required". I quickly pencilled a thumbnail sketch indicating approximate proportions for type, the restaurant logo, and room for a small illustration. I then proceeded to pick a typeface and size appropriate for the ad and submitted the copy to typesetting. As soon as I returned to my desk, the phone rang: the client wanted a proof in a couple of hours

It was time to physically build the ad. I went to my drafting table, pulled a fresh artboard from the shelf and with rulers and set squares, drew the borders. I wandered into the next room where mounds of various border tapes were kept, pondered the thickness of tape that would suit the ad, and travelled back to the drafting table. Although these tapes are fast and more consistent than technical pens, they have the annoying habit of bowing slightly when pressed into place. As soon as I sat down however, I realized that I needed an X-acto knife. (A number of bodies in the production department resulted in things being deposited in any number of unusual locations.) After finding everything I required, I continued with the ad. By this time the type was out and proofread: I waxed it and put it in place. The restaurant logo and an illustration were still required. I decided on a cheery little illustration of a couple dining. "Okay, so far so good" I exhaled to a nearby co-worker. "Now all I need is a couple of photostats."

Having worked with both darkroom and daylight cameras I can honestly say both are equally frustrating. Although the computerized daylight camera I've recently worked with is simple to operate, its extreme sensitivity makes it impractical for a large amount of work. After I figured out the percentage of reduction for the illustration and logo needed, I took several shots. Several is a common amount. Imbalances in chemistry, fluctuating temperature and humidity and many other factors make a good first shot a rare treat. Finally, the stats were done, waxed, cut and pasted up. The ad was complete.

I suppose it was relatively painless. There was a nagging doubt, however, that there must be an easier way. With a heavy workload and pressing deadlines, the process just didn't seem to be efficient enough to keep up with demands.

In the midst of all this soul searching I had a long talk with my friend and former fellow employee Derek Grime. On an impulse I proposed that I work with him at Beyond Graphics, a desktop video/graphics studio. "It's different you know" he said rather ominously. "It's a really big transition". Oh please, I thought, how different could it be. "Well... it's nothing like what you're used to dealing with". If I wasn't completely convinced before, that statement made me certain I was doing the right thing. I quit my job and joined Beyond Graphics.

Three weeks ago I was introduced to Professional Page and Deluxe Paint on the Amiga. Derek quickly briefed me on the available tools and effects. I started to experiment, and to my astonishment did things in minutes that used to take hours. I was thrilled. I was astounded. Then I got angry. Why isn't everyone using an Amiga? Whether you use it for animation, illustration, desktop publishing or music applications, it is an invaluable tool. I hope more people in my former situation will change their minds after seeing the Amiga in action. It may be a slow transition, but I have faith that the Amiga's reasonable price and user friendliness will convince even the most diehard technophobes to convert. You're probably shaking your head at this article and moaning "another testimonial from another late bloomer", but surely there is room in the computer clique for both technical computer whizzes as well as every day professionals who have brought their trades to the Amiga. Now that the physical aspect of creating something has been facilitated, my creativity has been unleashed.

Let's make a deal. You continue to technically upgrade yourselves and write great software and I'll go out and convert the creative computer-unaware public. Together I'm sure we can conquer the world.

Library Name And Management Macros

An annotated compendium of assembler library macros

by Eugene P. Mortimore

Eugene P. Mortimore is the author of the Sybex Amiga Programmers' Handbooks. He is currently a freelance consultant, providing customized solutions for the Amiga in the fields of programming, graphics, video, desktop publishing, and music. He also does software customization, hardware interfacing, and digital communications work for MS-DOS systems.

One of the things you quickly learn when you try to read a piece of Amiga assembly language coding - especially Commodore-generated coding - is that many of the assembly language routines are written in a 'shorthand' notation using assembly language macros.

These macros can save the assembly language programmer significant work, because they allow assembly language notation to be entered more conveniently. However, unless you understand how the individual macros are defined and what they are intended to do, you will find yourself groping to try to understand someone else's assembly language listing.

The purpose of this article is to highlight one specific group of related macros: the library name and library management macros. These macros appear time and again in the Commodore Amiga assembly language listings, and their definitions appear in many locations scattered throughout the Amiga assembly language INCLUDE files. Unless you understand the purpose and see the definitions of all of these macros in one location, it is difficult to see how they are related to each other and how they should be used to best advantage.

Once you understand the internal unity of these two macro groups, you will be able to read other people's assembly listings more easily. You will also be able to write your own assembly source much more quickly by taking advantage of these convenient shorthand macro notation.

The library-name macros

The table at the end of this article summarizes library name and management macros arranged in three different categories. The first category includes the basic library and device name macros. These macros were created to provide a shorthand notation for naming some of the Amiga libraries - both general libraries and device libraries - so that you can refer to the libraries more easily when you issue an OpenLibrary or CloseLibrary function call in your program. Also, any program that uses these library-name macros won't need to be changed if the library or device library name in the Amiga INCLUDE files is changed (though this is very unlikely, the precaution is still sound), since just the macro definition itself will change.

All of these macros are simple. They consist merely of the macro name ("AUDIONAME" for example) and a null-terminated string - a string of ASCII characters terminated with a zero byte, which is the library name itself. A typical example of this type of macro is:

```
AUDIONAME MACRO
          DC.B 'audio.device',0
          ENDM
```

Some of these library-name macros are defined using the above (or a similar) representation directly; others use the STRING macro (see the "exec/strings.i" INCLUDE file):

```
STRING    MACRO
          DC.B \1
          DC.B 0
          CNOP 0,2
          ENDM
```

(Here the CNOP assembler macro directive is used to force the characters in the STRING macro to an even-word boundary.) A typical example of this second type of name macro is:

```
PARALLELNAME   MACRO
               STRING 'parallel.device',0
               ENDM
```

These different representations for assembly language macro definitions - in this case, macros invoking macros - begin to point up some of the annoying peculiarities in the Amiga INCLUDE file macro definitions, and demonstrate how the INCLUDE files evolved over a period of time as the system programmers kept looking for quicker ways - more cryptic macro representations - to save themselves work.

The library management macros

The second and third parts of the table listeighteen macros that are devoted to general library and device library management.

Several of these provide examples of macros invoking other macros with specific values for arguments. For example, the CALLEXE macro invokes the CALLLIB macro with a new embedded definition for its one argument. The LINKEXE macro invokes the LINKLIB macro similarly. The ABORTIO and BEGINIO macros invoke the LINKLIB macro with specific embedded instances of their two arguments.

The macros usi a fairly standard assembly language macro notation system; that notation is currently supported by the Manx assembler and other up-to-date Amiga assemblers. Let's take a look at some aspects of that notation, especially as used in the presentation of this article.

All macro names - for example, CALLLIB, LINKLIB - and macro assembler directives are in upper-case characters. The macro directives of interest include:

MACRO	begin macro definition
EQU	assign label to value of expression
IFC	compare two strings for equality and assemble next assembly language lines if equal
IFNC	compare two strings for equality and do not assemble next assembly language lines if equal
IFGT	compare absolute expression to zero and if greater than zero then assemble subsequent lines
DS.B	define byte-data storage
DS.W	define word-data storage
ENDC	end of conditional macro clause
ENDM	end of macro
SET	set symbol or label to value of absolute expression
XREF	cross-reference a global program-wide symbol

Another directive used in macros is FAIL, which causes the macro to fail if the programmer supplies too macro arguments when he invokes the macro in his assembly language routine. Macros that use FAIL usually contain a IFGT test on the actual number of macro arguments in the line just above the FAIL directive. See the CALLLIB and LINKLIB macros for examples.

The macro symbol NARG is a special assembler symbol that always represents the number of arguments actually supplied when the macro was invoked. CALLLIB and LINKLIB provide examples of macro decisions based on NARG.

Dummy macro arguments

Like C functions, these macros take dummy arguments. Unlike C functions, though, the dummy macro arguments are optional and are only given as comments on the macro definition line itself (after the asterisk), not as positional arguments. The first dummy argument in the body of the macro's definition is referred to as \1. The second is referred to as \2, and so on.

If the slash characters are enclosed in single quote (as in '\1'), the argument is treated as a string of characters. If the quotes are not present, the argument is treated as a numeric variable and the current value of that variable is being referenced. The LIBINIT macro shows an example of both of these uses.

In both cases, when the macro is actually invoked in an assembler routine, the actual argument value on the macro invocation line replaces the dummy argument symbol either as a string or as a numeric value in the macro's definition.

The macros explained

In this section we discuss each of the general library and device library macros separately. These macros are mostly used to link an assembly-langauge program into the Amiga shared libraries, and access the general and device-specific library function vectors in them. You should consult some of your Amiga assembly language listings to see how these macros are actually used in typical programs.

```
CALLLIB   MACRO * functionOffset
          IFGT NARG - 1
               FAIL !!! too many arguments !!!
          ENDC
          jsr \1(A6)
          ENDM
```

The purpose of this macro is to provide a convenient way to jump to a specified function vector in an already-opened Amiga shared library. CALLLIB expects one argument - the function vector offset name; if the number of arguments supplied is greater than one, the macro fails. The characters after the FAIL directive itself are ignored by the assembler.

Otherwise, the macro causes a program jump to the library function vector specified by the macro's argument - the jump vector offset name. The function vector offset relative to the library base pointer - a pointer to an Exec-style Library structure - that the programmer has previously placed in register A6 after an OpenLibrary function call.

```
LINKLIB   MACRO * functionOffset,libraryBase
          IFGT NARG - 2
               FAIL !!! too many arguments !!!
          ENDC
          move.l A6,(-SP)
          move.l \2,A6
          CALLLIB \1
          move.l (SP)+,A6
          ENDM
```

The purpose of this macro is to link the programmer's code into an Amiga shared library and to call a function in that library. It expects two arguments: the first is the function vector offset name, the same argument used by the CALLLIB macro. The second argument is a pointer to the library base - a pointer to an Exec-style Library structure - for the particular library of

interest. Note that the macro arranged so that the original contents of A6 is undisturbed.

Note that LINKLIB is similar to CALLLIB macro except that it does more and requires two arguments. LINKLIB will fail if the number of arguments actually specified when the macro is invoked is greater than two.

```
LIBDEF    MACRO * libraryFunctionSymbol
\1        EQU COUNT_LIB
COUNT_LIB SET COUNT_LIB-LIB_VECTSIZE
          ENDM
```

This is a library definition macro. Its purpose is to define each library function vector offset name in the function vector offset table for a user-defined shared library. You would only use this macro if you were designing and defining your own Amiga shared library.

The macro expects one argument - an Amiga shared library function vector offset name. It equates that argument to the COUNT_LIB variable, then reduces COUNT_LIB by the LIB_VECTSIZE constant, the size of library jump vectors, normally six bytes. COUNT_LIB and LIB_VECTSIZE are initialized in the "libraries/dos_lib.i" INCLUDE file.

```
LIBINIT   MACRO * [baseOffset]
          IFC '\1',''
COUNT_LIB      SET LIB_USERDEF
          ENDC
          IFNC '\1',''
COUNT_LIB      SET \1
          ENDC
          ENDM
```

This is also a library definition macro. Its purpose is to set the library base offset - the negative offset from the library base address - from which library function vector offsets begin in a user-defined Amiga shared library. This library base offset leaves room in the library function vector offset table for the reserved function vectors in the library. Again, you would only use this macro if you were designing your own shared library.

LIBINIT expects one argument - the name of the library base offset for a user-defined library. The macro first tests to see if a non-NULL argument value has been supplied. It does this by testing the actual argument string characters against the NULL string using the assembler's IFC directive.

If no macro argument has been supplied, the COUNT_LIB variable is set equal to the LIB_USERDEF constant. On the other hand, if a non-NULL argument is supplied, COUNT_LIB is set equal to the supplied argument value. LIB_USERDEF is defined in the "exec/libraries.i" INCLUDE file.

```
CALLSYS   MACRO * functionOffset
          CALLLIB _LVO\1
          ENDM
```

This macro is not in the current version of the standard include files, but is commonly used by programmers. Its purpose is to call a library function at a specified vector offset name location in the current library. The only argument required is the name of the library function to be called.

CALLSYS is an extension extensions of the CALLLIB macro. It merely relieves the programmer of the tedium of continually writing the "_LVO" - "Library Vector Offset" - characters.

```
LINKSYS   MACRO * functionOffset,libraryBase
          LINKLIB _LVO\1,\2
          ENDM

LINKEXE   MACRO * functionOffset
          LINKLIB _LVO\1,_SysBase
          ENDM

LINKGFX   MACRO * functionOffset
          LINKLIB _LVO\1,_GfxBase
          ENDM
```

Again, these are commonly-used macros not found in the current generation of the standard INCLUDES. LINKSYS is a variant of LINKLIB (which it calls) that, like CALLSYS, adds the "_LVO" prefix.

LINKEXE and LINKGFX are both time-savers that may occasionally be useful. Both accept a function name as an argument, prepend the "_LVO", and call LINKLIB with a specific library base address, with which the relevant variable name (either "_SysBase" or "_GfxBase") must have been initialized.

```
LIBENT    MACRO * ftnVtrOstName
_LVO\1    EQU count
count     SET count-vsize
          ENDM
```

This macro sets the negative offset into a library's function vector table at which a particular DOS library function is located. The macro expects one argument - a library function vector offset name - which is equates to the *count* macro variable; it then reduces *count* by the *vsize* constant, representing the size of vectors in a library, normally six bytes. Note that LIBENT is identical to LIBDEF. The "count" and "vsize" variables are initialized in the "libraries/dos_lib.i" INCLUDE file.

```
EXTERN_LIBMACRO * functionName
          XREF _LVO\1
          ENDM
```

The purpose of this macro, which is sometimes declared under such other names as XLIB and XREF_EXE, is to define a function vector name as a program external global symbol. The macro saves the programmer from the tedium of continually writing the "_LVO" characters in his program's external symbol declarations (and may help in adding clarity to his code). In this sense, its relationship to the XREF directive is very simi-

lar to the relationship between the CALLSYS and LINKSYS macros.

```
DEVINIT   MACRO * [functionOffset]
          IFC '\1',''
CMD_COUNT      SET CMD_NONSTD
          ENDC
          IFNC '\1',''
CMD_COUNT      SET \1
          ENDC
          ENDM
```

This device library definition macro is very similar to the general library definition LIBINIT macro. Its purpose is to set the device library base offset - the number of bytes away from the library base address - that device library function vector offsets begin in a user-defined Amiga shared device library. This offset leaves room in the vector table for the standard device library function vectors-- that is, the standard device commands - in the device library. Once again, you would only use this macro if you were designing your own Amiga shared device library.

If the argument string is NULL, DEVINIT sets the CMD_COUNT macro variable to CMD_NONSTD - the number of non-standard commands in that particular device library. If the argument string is not NULL, the macro sets CMD_COUNT to the macro argument string itself. CMD_NONSTD" is initialized in the "exec/io.i" INCLUDE file.

```
DEVCMD    MACRO * cmdname
\1        EQU CMD_COUNT
CMD_COUNT SET CMD_COUNT + 1
          ENDM
```

This macro maintains a count of the number of total device commands in a device library. The standard device commands start with the CMD_RESET command and end with the CMD_FLUSH command. Study the "exec/io.i" INCLUDE file to see how the DEVINIT and DEVCMD macros are used.

DEVCMD takes one argument, the name of a particular device library command. It equates that argument to the value of the CMD_COUNT variable, then increments CMD_COUNT by one.

```
BEGINIO   MACRO
          LINKLIB DEV_BEGINIO, IO_DEVICE(A1)
          ENDM
```

This macro links the programmer's assembly language code into the BeginIO device library function entry point - that is, it calls the BeginIO function. Note that BEGINIO invokes the LINKLIB macro with two specific macro arguments. This is just another example of a macro invoking another macro.

```
ABORTIO   MACRO
          LINKLIB DEV_ABORTIO, IO_DEVICE(A1)
          ENDM
```

This macro links the programmer's assembly language code into the AbortIO device library function entry point - that is, it calls the AbortIO function. Just like BEGINIO, ABORTIO invokes LINKLIB with two specific macro arguments.

Conclusion

Judicious application of these useful macros can save you a lot of work. I hope that their presentation here, in one location, will help you understand and exploit them more fully. You could also use them as models for creating usefule macros of your own, a practice which will help in simplifying and streamlining your assembly language programming.

Macro Name	Include File Name	Args	Description of Macro
General Library Name Macros			
EXPANSIONNAME	expansion.i	0	Name of Expansion Library
AUDIONAME	audio.i	0	Name of Audio Device Library
CIAANAME	cia.i	0	Name of CIAA Resource Library
CIABNAME	cia.i	0	Name of CIAB Resource Library
DISKNAME	resources/disk.i	0	Name of Disk Resource Library
DOSNAME	libraries/dos.i	0	Name of DOS Library
EXECNAME	execname.i	0	Name of Exec Library
ICONNAME	workbenck/icon.i	0	Name of Workbench Library
MISCNAME	resources/misc.i	0	Name of Miscellaneous Resource Library
PARALLELNAME	parallel.i	0	Name of Parallel Device Library
POTGONAME	resources/potgo.i	0	Name of Potgo Resource Library
ROMBOOT_NAME	romboot_base.i	0	Name of RomBoot Library
SERIALNAME	serial.i	0	Name of Serial Device Library
TD_NAME	trackdisk.i	0	Name of Trackdisk Device Library
TIMERNAME	timer.i	0	Name of Timer Device Library
General Library Management Macros			
CALLLIB	exec/libraries.i	2	Call library functions
LINKLIB	exec/libraries.i	2	Link to a library, call a function
LIBDEF	exec/libraries.i	1	Define library function vector offset
LIBINIT	exec/libraries.i	1	Initialize library vector table offset
CALLSYS	(not standard)	1	Call a function without using "_LVO"
LINKSYS	(not standard)	1	Link to a library without using "_LVO"
LIBENT	libraries/dos_lib.i	1	Define offsets for DOS function names
EXTERN_LIB	exec/types.i	1	Define an external program symbol
XLIB	(not standard)	1	Another name EXTERN_LIB
XREF_EXE	(not standard)	1	Another name EXTERN_LIB
XREF_GFX	(not standard)	1	Another name EXTERN_LIB
CALLEXE	(not standard)	1	Adds "_LVO" to CALLLIB invocation
LINKEXE	(not standard)	1	Invokes LINKLIB using Exec Library
LINKGFX	(not standard)	1	Invokes LINKLIB using Graphics Library
Device Library Management Macros			
DEVINIT	exec/io.i	1	Inititialize device library vector table offset
DEVCMD	exec/io.i	1	Counts device library commands
BEGINIO	exec/io.i	0	Links to device library BeginIO through LINKLIB
ABORTIO	exec/io.i	0	Links to device library AbortIO through LINKLIB

Table of Library Name and Library Management Macros

Comic Setter

A new program for would-be cartoonists

Review by Jay Gross

Comic Setter: $99.95 US
Comic Art: Science Fiction, Super Heroes, Funny Figures: $34.95 US each
From Gold Disk Inc., P.O. Box 789, Streetsville, Mississauga, Ontario, Canada L5M 2C2

I wanted to be a lawyer. My family convinced me there wasn't any money in it. Then I wanted to be an architect. Nah, they said. Concert pianist? Writer? Ah! Cartoonist! Well, they didn't have to talk me out of cartooning. I had zip for talent, so that was that.

Nevertheless, I've finally made it. Now, with my Amiga and a new software product called Comic Setter, my dreams are a reality. I'm a cartoonist! I still can't draw a straight line with a ruler (there went the career in architecting). Oh, but now I don't *have* to know anything about art. Comic Setter does all the work, and I take all the credit. (I also take cash, money orders...)

Enough about me. This isn't a review of me - no wisecracks, please! What this is, is a review of Comic Setter, which is what makes the likes of me into a cartoonist, and with a lot less effort (on my part) than it took to make me a magazine writer. Comic Setter isn't a playtoy. It's a serious, professional-level product that happens to facilitate something that's out-and-out fun. Comics!

First the Panels

Like the comic books and comic strips it creates, Comic Setter works in panels. These can be as large as a page, or as small as you're willing to attempt. To help you set up your page, the program has a handy feature called "AutoPanel" which "autopanels" the page, all perfectly aligned and correctly spaced, instantly. Nice.

Even if you let AutoPanel set up the page, the structure is not by any means set in stone (engraved on screen, in this case). You can move, resize, or reshape the automatic panels to suit yourself, and although they must always be square or rectangular, they don't have to be visible, so you can do pretty much as you please with them. Since the program also lets you have stuff overlap into the spaces between panels, there really isn't much of a limit on what you can do.

Adding Artwork

Panels done? Put something in them.

If you can draw, you can use Comic Setter's very powerful drawing tools to make your own panels. Comic Setter provides kits of both bitmapped and structured drawing tools. The two types of drawings happily coexist in the same page at the same time, and if you like, you can collapse panels of structured graphics into a single bitmap. The program also permits you to "group" structured objects, which will then be considered a single object for further operations. You can assemble a graphic from several parts and then move, resize, crop, clone, or flip it all at once. Nice.

The program's curve function is particularly good for getting those ever so popular "S" shapes. If Comic Setter's Bezier won't draw the curve, *don't* ask her out! Of course, the program has the usual tools, but because they create structured objects, these have the advantage of being resizable - proportionally or not, your choice - to whatever shape you really wanted in the first place. The resulting image won't lose any detail. Comic Setter also permits resizing of bitmapped images, but of course they acquire the usual pixelization problems when you do drastic things to them.

The program's bitmap tools include the usual stuff, plus a nice spray paint tool (in case you need to redecorate any subways), and a "smear" tool that lives up to its name. Menu options permit flipping objects, either structured or bitmapped, vertically or horizontally. In addition, you can "copy" "paste" "duplicate" and other things most welcome in a cartoonist's environment.

To get text into a panel, you can create a bitmap graphic (which then behaves as a bitmap, and can't be edited as text anymore), or add a structured graphic box, which permits easy

"If anybody ever asks you what Professional Page is, the answer is: a printer driver for Comic Setter!"

editing and refining of both the text and the graphic elements that go with it - such things as border color, background color, text color, font, size and shape.

Comic heroes, villains, innocent bystanders, extras, rocks, and trees all talk by means of speech balloons, which are often shaped to match the mood, importance, or content of what they're saying. Same way with Comic Setter. You set up a speech balloon with a handy mouseclick menu, then place it where you want it. You can reshape, resize, and reword it as much as you like. Everything about speech balloons is controllable, or you can just let the program do its thing. The neatest part is drawing out the "tail" of the speech balloon toward the character whose utterance it contains.

For speech balloons ("cloud" ones, too, if your characters "think"), for text boxes, and for "to be continued..." notations, Comic Setter comes with several special, comic-esque fonts to augment the regular Amiga ones. The comic fonts are all rather small, although you can get larger lettering by resizing the text boxes. However, if you go too far with the resizing, you end up with blocky letters that look like you borrowed them from a stair builder.

If you're the same level of artist that I am (none), you'll have to resort to importing Amiga IFF files into Comic Setter to get something besides stick figures and geometric shapes in your comics. This is where Comic Setter makes cartoonists out of would-a-been's. That's "would-a-been a cartoonist if I could-a-been."

The Comic Setter package includes one disk full of general-purpose clipart. In addition, the company markets three (so far) additional packages of comic clip art, each of which contains around 70 illustrations on two Amiga disks. The extra clip art is thematic. There's a package for comic book figures, science fiction (excellent monsters!), and superheroes. The vast collection includes backgrounds, planets, spaceships, male and female characters, props, doodads, and body parts (to assemble into characters, or maybe for that "axe murder" comic you were planning?). You also get animals, supervillians (like superheroes, only *mean*), and the ubiquitous "sound effects" - nicely rendered traditional comic-style sounds like "Biff!" "Kertwaaaang!" and my fave rave, "ARRRGGHH!"

Comic Setter's import facility will work on any Amiga IFF graphic, but the Comic Setter ones are already matched to the program's default palette - a thoughtful touch for us armchair cartoonists. The Comic Setter clipart - IFF, all of it - will also work in other Amiga paint programs. Isn't the Amiga wondrous!

The program works in 16 colors. You can choose them, change them, or use fewer (but not more) any time you like. I found the default, richly colorful palette to be very appropriate to comic art. For the illustration which accompanies this review, which is in black and white, I should have remapped the palette to 16 shades of gray. I didn't.

Comic Setter works in the Amiga's high resolution (better known as "flicker") mode. You can change to medium res, if you wish. I prefer using the program in hi-res, but I have a very fancy monitor to do it with.

Real comics and funny papers are generally low-tech: paper. So, to get your comics from the high tech clutches of your Amiga onto low tech paper, you have to print them. Comic Setter prints to any printer that the Amiga's Preferences program knows about. It even offers an excellent interface to the finer points of graphics printing, independent of the Preferences program. This gives you control over Halftone mode, Color/B&W, and other printing decisions from within Comic Setter.

Although Comic Setter happily prints colour to colour printers, it doesn't support colour PostScript, or even PostScript, printing at all. I was really disappointed to see no support for PostScript. The product is otherwise very much a professional tool. A business, an ad agency, or a commercial artist is going to want PostScript output. Colour PostScript hasn't found its niche yet, since the printers cost $25,000, but it's coming, and Comic Setter cries for colour PostScript.

To get Comic Setter's output into PostScript, Gold Disk, makers of both Comic Setter and Professional Page, advises importing Comic Setter bitmaps into Professional Page. Indeed, that's exactly how the Comic Setter comics got onto the pages of this magazine. So, if anybody ever asks you what is Professional Page, the answer is: a printer driver for Comic Setter!

One of the amazing things about the program is how neatly and well it manages memory. At 300K of executable code, Comic Setter is a BIG program. It won't work in a 512K Amiga. However, it does work in a memory-starved environment - an expanded memory Amiga with a bunch of ram occupied by other things. If you starve the program's memory needs, it'll switch its screen to black and white while it's doing its work, but it *will* work. In addition, the program provides a menu option to swap pages to and from disk. This permits multiple documents (selectable by a cleverly dynamic pull-down menu) of multiple pages to be worked on at once. You can cut and paste between pages, or even documents, in this manner. This virtual memory management scheme would be *very* welcome in Professional Page.

I was not able to crash Comic Setter by multitasking other things behind it. I tried. That says all that needs saying about

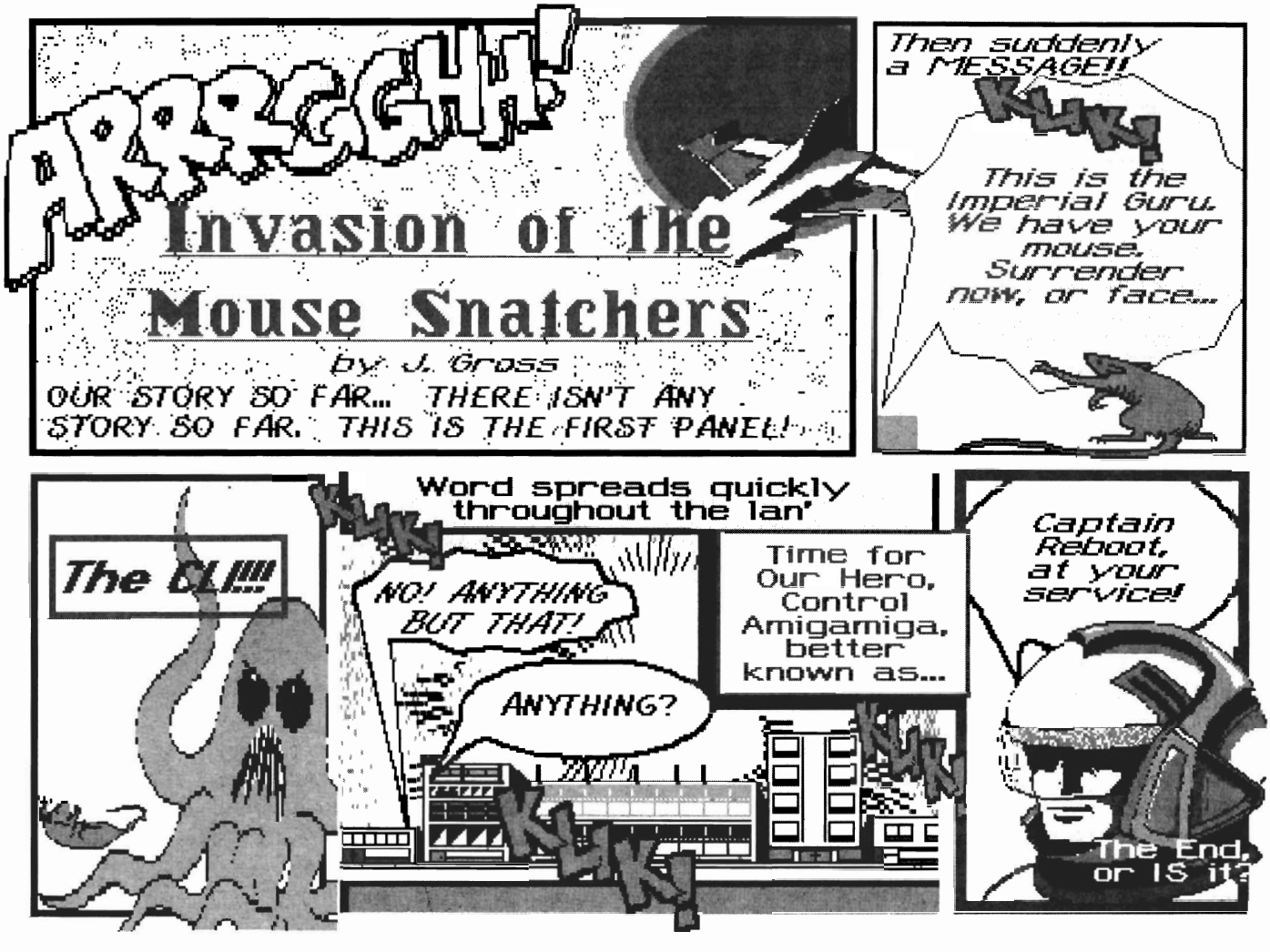
how well Comic Setter multitasks. I had FACC-II, GRABBiT, WorkBench, ClickToFront, FastFonts, ScreenBlanker (from MicroSmiths), a ramdisk (not RAD:), ASDG's recoverable ramdisk (not RAD:!), Snipit, and an open AmigaDOS 1.3 Shell with about a dozen C: commands (not ARP) resident and... well, you get the idea. All this in a 1.5-meg Amiga 1000 left about 910,000 bytes for Comic Setter under KickStart and WorkBench 1.3. Comic Setter performed flawlessly.

The Wish List

Nice as it is, Comic Setter does have a few minor shortcomings (doffing Cartoonist's hat, switching into critic mode). For one thing, it could really use some additional comic-style fonts, and larger sizes of all of them, even if they were sold as a separate add-on package. The Amiga's system fonts are too, too, *plain* for us high-level cartoonists.

Second, it would be nice to be able to twist, rotate, shear, bend and otherwise distort bitmaps or objects while creating comic panels. Of course, these effects can easily by achieved with any of several other Amiga paint packages, and the results can be imported, but it'd be nice to have these things at the fingertips within Comic Setter. Conservative as it is, Comic Setter is too big to multitask with memory hogs that offer this feature. Actually, maybe chip memory itself is too *small*, eh? Is anybody listening, out there? West Chester? Calling West Chester...

And third, it would be really, really nice if Comic Setter saved in formats which are acceptable to Professional Page and also to PostScript printers. Since Gold Disk is threatening to ship a professional, structured drawing product imminently, it'd be nice if Comic Setter and Professional Draw shared a graphics format, although it's not likely.



This comic panel was created with Comic Setter. All of the art here came from the "General Clip art" collection included with the product, except for a few circles, lines, and filled rectangles. The scatter background in Panel 1 was done with Comic Setter's spray paint tool. The finished comic was saved to an Amiga IFF bitmap at 608 x 450 pixels, which was then imported directly to Professional Page, resized slightly, and printed to Postscript. The largest of the fonts ("Invasion...") is a ported Mac Font. The rest come with Comic Setter. Total production time was several hours, but it was *fun*!

"I was not able to crash Comic Setter. I tried. That says all that needs saying about how well Comic Setter Multitasks."

Another big wish is for structured objects to import into the program. It would-a-been nice if at least some of the Comic Setter clip art had been structured, so it could be resized with impunity.

While still in wish mode, might as well mention Comic Setter's patterns. They're okay, but not thrilling. The biggest problem with them is they're not comic-book-ish. Comic Setter gives you brick walls and stuff, but no big, dramatic dot patterns like in the funny papers.

Problems

After Comic Setter survived my best efforts to crash it, successfully printed to my printer, and saved its files to both its own format and to Amiga IFF, I gave up trying to find things out-and-out wrong with the program. There are a few small quibbles I have with it, however. Minor things, all, but things, just the same.

The first thing is, Comic Setter's panel saving. It saves panels as Amiga bitmap IFF files, and the resulting pictures are suitable for uploading to BBS's, importing to other art programs, or whatever. The problem is that each of the panels is saved in a format appropriate to its size and shape. A small panel is saved in low-res, and a large one is saved in high-res. Medium ones... you guess it, medium-res. A three-panel comic strip might save in three different resolutions. Panel saving really needs to be the same resolution throughout, perhaps with a menu option to either match them or perform the "best fit" operation the program currently does.

The next small problem has to do with the program's bitmap save for an entire page. If the page is the usual 8.5 x 11 or so "letter" size, the resulting Comic Setter IFF bitmap is an enormous 792x1008 pixels. Very few Amiga software products can deal successfully with that image.

The only other real problem seems to be the extremely sensitive nature of the control points for its boxes, panels, and drawings. This is probably partly a function of the high-res mode in which I prefer to work. The pointer positioning required to activate any of the control points is extremely critical, requiring very careful mouse motion.

Summary

Comic Setter is a nice program. After all this raving, were you expecting a crummy summary? Nothing crummy about it. Comic Setter is fun to use. It's extremely easy to learn, and it's productive, almost from the moment you start. You don't have to be a professional *anything* to use the program, least of all a professional artist or cartoonist. You don't even have to use the program to create comics!

The program does require some learning and just a bit of poring through the manual, but with only a few minutes invested, you can get going strong, maybe turn out some one-of-a-kind creations to torture your relatives with. Or, you can get serious. Explore the intricacies of the program's graphics tools, and turn out stuff that Mad would be proud of. Or maybe Marvel?

Comic Setter lists for $99.95 in the US, $129.95 in Canada. Frankly, that's much too cheap for a piece of software of this quality and productivity, but don't anybody tell Gold Disk, okay?

Okay, this is all the reviewing you're going to get. I need to do another comic for the occasion of my mom's birthday. Now where is that clip art...?

AmiWare

Arts Education Entertainment

3640 Weston Road, Weston, Ontario, Canada M9L1W2

	Cdn.$
ARexx	44.00
C.A.P.E. 68K	87.50
CygnusEd Professional	92.00
Lattice C v5.0 Developer	290.00
Lattice C++	400.00
Manx Aztec C Developer	299.00
Modula 2 Developer (TDI Software)	147.00
Multi-Forth (Creative Solutions)	87.00
Pascal 2.0 (Metacomco)	125.00
Power Windows 2.5	87.50

If you are looking for a product that is not listed, give us a call.

Prices are subject to change without notice

Orders (416) 737-7385
BBS (416) 770-0124

Ultra DOS Utilities - Module 1

Review by H. Russel Douglas

Ultra DOS Utilities Module1: DOS utility system
From Free Spirit Software, Inc.,
905 W. Hillgrove, Suite 6, La Grange, IL 60525
Suggested retail: $59.95 US

Let me begin by saying that this is a useful product for the intended audience. It is aimed at the intermediate to advanced Workbench user. It provides a well thought out interface and a variety of very useful functions.

The user interface is through a full screen control panel. It is well laid out and provides an obvious navigation system to the Amiga file structure.

It would be easy to overlook the value of this product. The packaging is not slick and the first impression is one of a mediocre product. This is more a comment on our expectations of marketing than anything else. We expect the packaging to reflect in some way the value of the contents. How often have we been disappointed in what we have received in a slick package. *Ultra DOS Utilities* delivers good value at a fair price.

The manual, like the packaging, is not slick. But while the packaging is adequate, the manual needs work. I will deal with the manual in detail later. It does not belabour the reader with long explanations but provides an adequate description of the product. A new Amiga user will be able to use the product with the manual, but should have a reference text on AmigaDOS available as well.

Ultra DOS Utilities perform a number of AmigaDOS related functions. These are presented to the user in one major screen divided into three functional areas.

Environmental Control Section

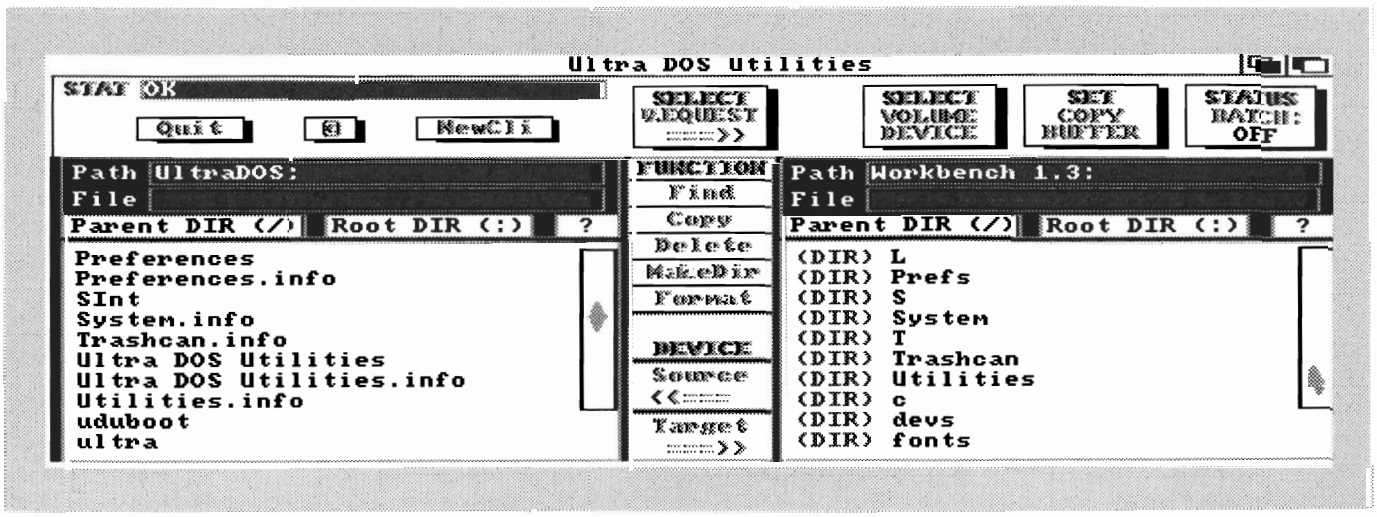
The top section of the screen is the Environmental Control Section, which allows control of the global systems functions. In this section the user can select one of two file display requesters. Selecting a volume or device will then assign that volume or device to the selected file display requester. The set copy buffer size and clear or set the batch mode work globally. An additional display provides status info for all operations.

File Requesters Section

The next section is divided into two file requesters, separated by the functions section. These requesters act as source and target for the various file manipulation features. Either requester can be selected as the source, the other one becomes the target by default.

The file requesters are similar in layout and allow the user to have two independent views into the system. Each requester has a number of components which allow easy navigation through the file system. The user can select the path and file of interest. Short cuts are provided by gadgets which allow selection of the parent directory or root. An additional requester is provided which provides detailed information about the file.

The info screen provides information similar to that provided by a list command issued from CLI. Two additional features



are included which prove to be useful. The first is the ability to invoke ED, the Amiga full screen editor, to edit a selected file. The second is the ability to view IFF files (note that HAM images are displayed in 32 colors only). No editing capability is provided for graphics images.

Functions Section

The third section is the functions section. Its placement is logical, as it is centered between the source and destination requesters. The functions section provides access to the file manipulation features of *Ultra DOS*.

At the bottom of the functions section are the source and target gadgets. These work together and determine which file requester will be the source and which will be the target of all operations. Clicking on either will change the other so that no conflict occurs.

A find function is included to perform an exhaustive search for a particular file. The search is case sensitive and wildcards are permitted.

The copy function allows standard copies as well as a batch file facility. The standard copy function allows copies of directories and or files to be made using both wildcards and date/time groups. The wildcards are non-standard but flexible nonetheless. The date/time facility allows copies of files before, during or after a certain date/time period. This allows the user to perform incremental backups for hard disk users.

The standard copy feature allows multiple copy options. In order to copy more than one file at a time, the copy function requires that the files be related by some naming convention or date/time. Many times we have files we wish to treat as a group which are not related in an obvious way. The batch file facility allows the copying of groups of files selected by the user. The user selects a file by marking it with the mouse. Once all the desired files are marked the batch copy feature will copy all files indicated.

A useful option of both copy features is the ability to select Autoformat. When Autoformat is selected *Ultra DOS* checks the size of the file to be copied, then checks the available room on the target disk. If there is insufficient room the user is prompted for a new disk, which is then formatted before the copy takes place.

The functions section also includes the MAKEDIR and FORMAT commands for complete disk management.

Overall impression

Overall this is a good product for the intended audience. It suffers from average packaging and poor documentation. These things do not detract from the utility of the product, as the user interface is intuitive and the documentation adequately explains the more subtle points.

However, *Ultra DOS Utilities* Module 1 seems either to have been rushed to market or produced on a very low budget. A low budget is no excuse for the state of the manual. If this is Module 1 as the name implies, it should be re-released before Module 2 is produced.

I have included a few suggested improvements that would make a re-release a very much more powerful and useful product. If these were added and the manual rewritten I believe Free Spirit could have a very good product.

Suggested improvements

First the manual should be rewritten. It serves this product very poorly. The format is inconsistent. The table of contents has a totally different numbering convention than the book itself. In fact, all of the page numbers in the table of contents are wrong.

The numbering convention within the book does not seem to make sense. The *Ultra DOS* control panel is introduced with a centered heading. It shows three basic sections. The first section labeled "Section 1. The File Display Requesters" is left justified. The second section labeled "Section 2. Environmental Control Section" is also left justified. The third section labeled "The Ultra DOS Utility Functions" is centered and does not refer to a section in the title. Within sections or subsections the numbering/lettering scheme is also inconsistent.

The content of the book is not bad but it appears to follow no single logical path. It would be better to have separate reference and tutorial sections. More examples of problems could be given, but the message is clear: rewrite the manual.

The product could use a few additional features that would add greatly to its value and probably justify a re-release.

Hard disks are becoming more pervasive and the volume of data that users wish to back-up is becoming much greater. *Ultra DOS* should provide more storage options suited to hard disk management. It should offer data compression to allow more efficient storage of files. This could be combined with a fast copy to speed up the long waits when backing up to floppy. Obviously a data expansion mode would have to be provided to allow restoration of data and files.

The Select Volume/Device gadget screen would be better if it paired the related volume and device names rather than presenting them as two disembodied lists.

It would be useful to be able to specify a different editor than ED. Many users have a favorite editor and to force them to use ED is a great disservice.

Finally, the product should be updated to AmigaDOS 1.3. I realize AmigaDOS 1.3 was not available when the product was first released, but it is now. *Ultra DOS* should incorporate the new features offered by 1.3.

PIXMATE

Image processing for the masses

Review by H. Russel Douglas

PixMate: Image processing software
From Progressive Peripherals and Software

PixMate is a well designed, well executed and well documented product. It makes a whole new area of technology - image processing - available to the Amiga user. Pixmate's user interface and documentation succeed in hiding the complexity of image processing from the neophyte user. For the sophisticated user, many advanced image processing features are available. Some of these features had only been available on dedicated image processing work-stations and minicomputers before the release of PixMate.

PixMate is one product which at first glance would seem to be applicable to only a very specialized group of users. It would seem to only be of use to those professionals with a need to perform sophisticated image manipulation. However, after having used the product for a number of weeks, I believe the product has utility for a number of diversified groups.

You might ask yourself, who could reasonably be expected to make use of image processing power that was only available to scientists and graphics labs a few years ago? Well, it turns out PixMate's combination of a good user interface and an UNDO command opens up sophisticated image processing to the general computer user. A list of potential users and uses that springs to mind:

People who prepare documentation could use it to improve image quality for page layouts. They could reduce the number of colours to make the images more easily printable. In addition, they could use PixMate to produce the colour separations required for printing.

Programmers could use the package to convert screens from one graphics mode to another. One reason to convert is to reduce the number of bit planes (colours) used in a picture to improve memory requirements.

People in the desktop video business could make extensive use of the product in the preparation of the pictures they use in production. They could use the size features, colour adjustment or the extensive editing features.

Sophisticated amateur video buffs could use it to improve the results they achieve with their video reproduction gear.

Computer artists could use it to experiment with many different effects on their original art - who knows what new creations might emerge? An artist could use PixMate to convert an image from one resolution or format to another in order to use a favorite graphics package.

People who like to play with graphics: the special effect possibilities and graphics manipulation possibilities possible with this product are just too good to pass up.

So what is all this talk of image processing? Well it stems from the idea that once an image is captured in a computer, we can utilize the processing power of the computer to manipulate the image data to achieve a number of effects. These effects range from the merely useful to the stunningly innovative.

What is it that is so amazing, or more importantly so useful, that a computer can do with an image? Rather than discuss in generic terms what a computer can do with an image, let us deal specifically with what your Amiga can do if equipped with PixMate.

Because image processing is largely an intuitive activity, PixMate offers a number of safeguards which make it very forgiving and a lot of fun to experiment with. First, there is an UNDO that works for just about any PixMate operation. Second, there is a second screen on which you can keep a copy of your most recent changes, if you are worried about the effect of a particular operation. If you begin an operation and discover part way through it is not what you desire, all you have to do is press the space bar to stop the operation and press UNDO to restore the screen.

PixMate is a very sophisticated product, and I cannot hope to cover the myriad of features in an article of this type. I will attempt to give you an overview of the types of image processing you can accomplish with this product. In almost all cases, each of the features I discuss will have many more options than I have space to present.

...perhaps unique is the ability to convert a low resolution image to HAM, making the image available for editing in a HAM paint program...

PixMate's features are arranged in five logically related groups. These are **File Management, Edit, Color, Info, and Effects.** The **Effects** section is by far the most complicated and interesting and I will cover it in the greatest detail.

Image File Management

In the management of files as in all areas PixMate offers much more than even the high standard of user interface normally provided on the Amiga. In each area of file management the creators of PixMate seem to have anticipated the one feature you found left out of the last package you used.

For one thing, the file requester presents all devices known to the system. Once you get to the directory you wish, PixMate offers you multiple options to display the contents. You can sort the entries in the directory alphabetically, by size, or by date. You can load and save files in various formats: standard and compressed IFF format, Raw images, old style Digi-View images. It will even load Atari Neochrome image files, but will not save them.

Edit Menu

PixMate has two complete screens. Each can hold separate images with their own palettes and resolutions. Each can be operated on independently as if PixMate was dealing with the image in isolation. Much of PixMate's power actually comes from its ability to combine the two images in unique ways. (I will cover this in more detail in the effects section.)

Within the current screen, you can disable the title bar and the sprite. This is useful if you wish to take screen pictures of the images you have created. You can clear part of the screen, or scroll the image around on the screen. Any portion of the image scrolling of the edge is lost and can not be recovered by UNDO. A further feature allows you to move the image around on the screen and center it again. This feature is useful in centering overscan images before editing them.

If you toggle to the 'other' screen all of these features apply equally. In addition you can perform some basic two screen operations in edit. You can copy an image to the other screen. You can copy part of an image to the other screen. Finally you can clear the other screen or kill it completely.

PixMate multitasks provided you have enough memory. The most interesting part of this multitasking is PixMate's ability to 'grab' other screens to work on as images. PixMate can also be requested to grab a DeluxePaint screen, ignoring any other screens that may be open.

Colour

PixMate has an incredibly rich set of features with which to deal with colours. Since PixMate is not a drawing package but rather an image processing package, you do not draw with colour, but adjust the colours of a created image. You deal with colours by means of the palette. All the changes that you make to the palette will affect the whole drawing.

You can select any colour in the palette and adjust it using the familiar red, green, and blue slide bars. You can COPY any colour to another position on the palette. This will change any portion of the picture that used the colour you just copied over. You can EXCHANGE any two colours, which changes the position of any two colours in the palette. You can SWAP any two colours, the mechanics of performing a SWAP are similar to EXCHANGE. However, once the colours are swapped on the palette the picture is re-coloured so that it looks the same.

In addition to adjusting the colours in the current image's palette, PixMate gives you many options for combining or changing palettes. You can load the palette of the other screen or another image file. You can also take the palette of the current picture and copy to the other screen or you can exchange the palette with the other screen.

You can modify all of the colours in the current picture by choosing the Colour Bias option. This requester allows the user to adjust all colours of the palette, simultaneously, by modifying one of the six constituent sliders. These sliders are RED, BLUE, GREEN, CONTRAST, SATURATION, and INTENSITY. Lowering the RED slider will reduce the red component of all colours equally.

You can reduce the number of colours in a picture, perhaps to make it reproduce better on a poster or use less memory in a computer. PixMate can convert an image from almost any resolution to any other. It deals with all resolutions from HAM-Interlace to two colour Low-Resolution. It also deals with the Extra-Half-Bright mode. Also perhaps unique is the ability to take a low resolution image and convert it to HAM, making the image available for editing in a HAM paint program.

Effects

This menu is deceptively simple, as it only shows only four menu items: Display, Image Processing, Histographic Equalizer, and Reformat As. These four choices offer ways to reformat or change your image in literally thousands of ways.

The display control window allows you to select which bitplanes of the image will be displayed. Clicking on the appropriate number toggles that bit plane on or off. You can also swap one bit plane for another, rotate the bit planes, or change screen resolution.

You can increase or decrease the size of an image in order to allow it to fit in a particular presentation: part of the picture or the whole thing can be expanded or reduced by a factor of two or three. The reduction can be done in a number of ways: by preserving even scan lines, odd scan lines, or averaging the lines to be combined.

The next menu item is the Image Processor Window. This is by far the most comprehensive and complex section of the program. It is broken into sections: Logical Operations, Pixel Operations, Matrix Operations, Slide Control Filters, Data Modes and Processing Modes. These operations allow you to combine the images from the two screens in a number of ways.

"PixMate succeeds in hiding the complex algorithms from the user - new users to the world of image processing will find their exploration fun.."

Logical Operations affect the entire screen and treat all pixels in the same manner. You can chose from the various logical operators such as AND, NOP, OR, EOR, and SUB or subtraction. In addition there is a NOT operator which can be combined with other operators (eg. NOT and OR yields NOR). The logical operators take the bit patterns of the various bit maps on the source images and combine them to produce a resultant combined image. For example, using the AND operator will combine the two screens and remove any parts of the current screen which do not match the other screen. The results are interesting, and there is always the UNDO if you get something you don't like.

Pixel Operations concern themselves with the examination of individual pixels. These pixels are taken and examined in relation to other pixels. Computations are performed and the pixel is replace after being modified by the results of the computation. The pixel operators are AVG (Average), RND (Randomizing), MF1 and MF2 (Median Filter), USM (UnSharp Masking), LCE (Local Contrast Enhancement), and BIN (Binary).

AVG tends to smooth or blur details and make an image appear more clear. RND makes an image more grainy. MF1 is used to remove small defects without seriously distorting major features. MF2 produces a stronger effect than MF1. USM results in a sharper, more detailed image. LCE is similar in effect to USM, but may produce better results in some circumstances. BIN will create a two-colour image of the input source.

Matrix operations allow the user to selectively enhance certain details in an image. There are seven different matrix operations, each biased to the enhancement of a different kind of edge. For example, the '|' matrix operation will enhance vertical edges, and the '*' matrix operation will enhance edges in all directions.

The Data Options allow the user to select whether the processed data replaces the current screen data or is merged with the original image using an overlay technique.

The Processing Modes selects which colour processing technique to utilize to arrive at the final image. The trade-off is speed and memory requirements versus accuracy. The highest speed, lowest accuracy option is Mono Mode and the lowest speed, most accurate mode is AutoHAM. AutoHAM take at least four times as long as Mono Mode.

All of these features are related; the effects of one operation can be passed on to another, which will modify the picture and pass the results on to the next. As a simple example, you could take two images as sources and OR them, pass them through MF1, and then utilize Matrix '+' before combining them again on the original screen. This would have the effect of combining the two images, removing small defects, then sharpening the horizontal and vertical edges before placing the resultant image on the original screen.

The final feature I will discuss is the Histographic Equalizer Window. The name itself sounds intimidating. It shows how colour information is distributed in an image. With this option you can selectively reduce the number of colours in an image or enhance contrast.

I have covered a number of the features available in PixMate. I have not dealt in any detail with the more sophisticated features for combining images. To do so would involve a detailed discussion of image processing techniques. Part of the fun of PixMate is that the user is encouraged to experiment.

The manual is extremely well done. It takes what is a very complex subject and provides a logical introduction that never leaves the new user feeling stupid. PixMate succeeds in hiding the complex algorithms from the user, and makes image processing fun. New users to the world of image processing will find their exploration fun.

If you already are a user of image processing technology, you will not be disappointed. PixMate has an incredible array of features, which are all dealt with in detail in the reference section of the manual. It will provide features you are used to seeing only on high cost dedicated image processing workstations.

I hope I have succeeded in whetting your appetite for the things that PixMate makes possible. When I received the program to evaluate I thought an image processing package was an intriguing idea. I enjoy technology for its own sake, so I thought the package would prove to be an interesting intellectual exercise. After using it for a few weeks I did not want to give it back. It was entertaining to play with, useful for modifying previously created art and a great tool for preparing images for the programs I write.

-MAX™ The Build It Yourself Hard Disk System

- **No more painful waiting for icons to come up or large programs toload. -MAX- can provide over a 1/2 megabyte/ second reading rate with v1.3 FastFile System!**
- **Supports many 8 and 16 bit controllers, 68000 family CPUs and works with virtually all ST506/412 compatible drives.**
- **Compatible with most internal and external expansion devices.**
- **Once constructed, drive configuration is very quick and very easy using of a simple ASC11 text file and utilities provided.**
- **Typical cost for a 16 Megabyte hard disk system around $400.00. (Based on used Hard Disk of $100)**

Features: 1. Configurable for 1 or 2 controllers with 1 or 2 drives/controller. 2. Supports hundreds of megabytes/ drive and multiple partitions. 3. Can attach to and pass the 86 pin exp. bus for A500/A1000, or configure for A2000/ ZORRO II or ZORRO I 100 pin bus. 4. Supports OMTI, Adaptec, Western Digital and other controllers.

Includes: 1. Manual with illustrations, layouts and schematics, vendor list where parts can be purchased, (as no parts are supplied). 2. Device drivers, example configuration and Mountlist files. 3. Sophisticated hard disk exerciser/formatter/debugger. 4. Utility program with filecheck, diskcheck and archive bit functions. 5. Electronic Horizons' full and incremental backup utility.

To Order:
V3.0 available direct for $169.00. Send Cashiers Check or Money Order, COD (no Credit cards). PA residents add 6% sales tax. Outside USA: Money Order payable in US funds only, add $10.00 S & H. Business hours from 10 am to 9pm EST Mon-Sat. For tech, inquiries, please call between 6pm & 9pm EST Mon - Fri; 10 am & 5 pm Sat.

PALOMAX, Inc
424 Moreboro Road
Hatboro, PA 19040
215-672-6815

*-MAX- is a trademark of **PALOMAX**, Inc.*

√√√√ UEDIT √√√√

Programmable Editor for Technical AMIGA Users

Uedit has quietly become the Editor Of Choice for bright programmers and professionals in the USA and more than a dozen countries. Recently it won the Gold Star in Commodore magazine.

FEATURES: Ultra power, capacity, and flexibility. **Uedit** has a **Rexx** interface, many wordprocessing features, optional spelling checker, and a truckload of editing features. How do you describe a program that lets you add new features when you feel like it? Yet you can use it right away "as is". Edit 20, even 100 or more files. Auto-load/edit/save a batch of files while you relax. Every command and feature is customizable, even the online Help and Teach Keys facility!

USER SURVEY: 37% have a doctorate in physics, CS, math, engineering, medicine, etc. 82% use **Uedit** for programming. 73% use it for wordprocessing. 75% use the standard config "as is" or modify it slightly. 25% configure **Uedit** to emulate a popular program or to create their preferred environment.

"Definitely the most sophisticated text editor to come along last year . . . easy for beginners . . . and advanced users . . . reviewing UEDIT is like an art critic reviewing a lump of clay — you can mold it to suit your needs . . ."
Commodore August 1988

Uedit is $59 + $3 shipping. (Foreigners add $3). It includes:
- * Fully customizable **Uedit** plus shareware version
- * Professional quality 173 page printed manual
- * $15 shareware commissions for finding new users
- * Mailings, notice of upgrades, user assistance

Options:
- * Spell-checking **Uedit** & dictionary utilities, $15
- * Quarterly Newsletter, $12 per year

Rick Stiles, P O Box 666, Washington, IN 47501 (812) 254-4986

Budgeteer
budget preparation software

- **EASY TO LEARN AND USE** . . . utilizes fully *AMIGA** windows and menus . . 40 page manual
- **SPREADSHEET-TYPE FORMAT** . . . intuitive point-and-click data entry . . . cash-based accounting
- **VERSATILE** . . . 68 budget categories, fully user-definable . . . compare actual to budgeted expenses for 1 or 2 years . . . forecast future net earnings
- **PRINTED REPORTS** . . . tables of actual budgeted expenses, credits, balance (by month, for any year)
- **WIDE RANGE OF COLOR GRAPHS** . . . 36 bar graphs of actual/budgeted credits, expenses, etc . . . compare 2 years' results . . . pie chart of yearly expenses
- **OTHER FEATURES** . . . graphic dump of any graph or table (to printers supporting this feature) . . attach comments to data . . . tables and graphs of cumulative expenses

Not copy-protected. Available 1 Feb 1989.
Distributor inquiries welcome.

Suggested List
39.95 (U.S.)

prakticon, inc.
the practical software people

27 Hutchings Ct N.E.
Medicine Hat, Alberta
Canada T1C 1G3

* Trademark of Commodore Business Machines

AMIGA RGB CABLES

CUSTOM RGB
AMIGA TO 4 BNC
SONY KV1311 23F-34F
SONY MULTISYNC 23F-9M
NEC MULTISYNC 23F-9M
MAGNAVOX ANALOG 23F-6MDIN
MITSUBISHI MULTISYNC 23F-25M
ATARI ANALOG ADAPTER 23F-13F

ALL
Shipped Blue USA
29.00 TO 39.00

Call **For Quotes On Weird or Unusual Cables**

Typical Non-Standard Cable

REDMOND CABLE

For The Dealer Nearest You!

EAST COAST **615-478-5760**
WEST COAST **206-882-2009**

AmigaTEX

A portable document processing system

by Eric Giguère

Radical Eye Software
Box 2081 Stanford, California
USA 94309

Voice: (415) 32-AMIGA
BBS : (415) 32-RADIO

Email: rokicki@polya.Stanford.EDU

Prices:

AmigaTEX	*$200*
AmigaMF	*$ 75*
Printer driver set	*$100*

Add $10 for shipping outside the U.S. All prices in U.S dollars.

A demonstration copy of AmigaTEX can be obtained by sending a blank disk to the above address. The demo may also be available through your local user's group.

When processing text by computer, the user must decide what type of software to use. Word processors are most common, formatting text using a page-by-page approach, while page layout software offers a visual approach to text processing.

And then there's AmigaTEX.

AmigaTEX is a portable document processing system for those who want typeset-quality output and complete flexibility. We're talking total control. AmigaTEX is overkill if all you want to do is write a few letters, but it's perfect for mathematics, tabular data, and long documents, especially books.

The TEX System

AmigaTEX is a complete implementation of Donald Knuth's TEX (pronounced as *tek*) document processing system. Knuth is a professor at Stanford University and a prolific writer. You might recognize him as the author of *The Art of Computer Programming*, by most accounts the definitive (and as yet unfinished) series of books on programming.

Knuth describes TEX as a "typesetting system intended for the creation of beautiful books". TEX is an actual language that describes the structure and layout of a document. TEX commands are embedded in the document much as in a word processor; the document is then passed through TEX to produce the output file, and a separate printer driver is used to print it out.

One of the important features of TEX is its portability: TEX was designed to be independent of both the computer it runs on and the printer it outputs to. When a TEX document is processed, a device-independent (DVI) file is produced. This DVI file can be printed on any system that supports TEX, and the output will be identical on all, within the limitations imposed by the printers. A journal, for example, might use a laser printer for in-house editing and layout and then ship the resulting DVI file to a printer for output on a high-density typesetter.

The TEX manual, by Knuth, is known as the *TEXbook* and is usually purchased separately from TEX software.

TEX is not in the public domain, but Knuth allows software developers to port it to different systems as long as they don't alter it. The American Mathematical Society owns the trademark for TEX - a TEX implementation must pass a rigorous set of tests devised by Knuth to be called TEX.

LATEX

One of TEX's most powerful features is its ability to define new commands. These macros are sequences of existing TEX commands and other macros, and can accept parameters. Macros are used for commonly-used TEX features, much like subroutines or functions in programming languages.

On its own TEX can be quite intimidating as the commands are fairly low-level. A set of pre-defined macros called LATEX was developed by Leslie Lamport to simplify TEX and to allow the user to concentrate on the structure of a document. LATEX provides the user with a set of style files which define formats for several types of documents. Complete descriptions of all LATEX commands and styles are found in Lamport's *LATEX User's Guide & Reference Manual*, which is also pur-

chased separately. LaTeX style files can be easily altered to create new document styles. "Raw" or "plain" TeX is always available underneath LaTeX as well. LaTeX will not run on a 512K Amiga, however.

LaTeX is included with most implementations of TeX that I know of, including AmigaTeX. For the remainder of this article I will use TeX to refer to both TeX and LaTeX interchangeably.

AmigaTeX

AmigaTeX is an implementation of TeX for the Amiga ported by Tom Rokicki and distributed by Radical Eye Software. AmigaTeX includes TeX, LaTeX, some standard TeX-related programs, a *Previewer* with a complete set of fonts, and a 186-page manual. The manual is in addition to the *TeXbook* and the LaTeX *Guide*, which must be purchased separately. AmigaTeX can run on a variety of machine configurations, including a 512K Amiga with two disk drives. At least 1 megabyte is recommended, however, for serious work. A hard disk is optional. I use AmigaTeX on an Amiga 2000 with 3 megabytes of memory and two disk drives.

AmigaTeX can seem quite intimidating at first because it arrives on 10 diskettes. Six of these diskettes, however, hold fonts for the *Previewer*. The remaining four diskettes hold the executables for TeX and the *Previewer*, the various macro definition (including LaTeX) and style files, and some sample documents. The basic TeX working files (TeX, LaTeX, the *Previewer*, CLI commands, style files and a text editor) can be placed on two diskettes with some room left over to store current documents. The manual gives complete instructions on how to organize the TeX files to suit your system. If you have more than 1MB of memory you might want to load TeX and the *Previewer* into memory simultaneously (perhaps in a recoverable RAM disk) under different CLI shells for ease of use plus access to the *Previewer*'s automatic tracking facility.

Since AmigaTeX is a full TeX implementation, let's look at what makes it different from other implementations.

Using AmigaTeX

Once you've configured AmigaTeX to work on your particular system, you're ready to produce your first documents. AmigaTeX assumes that you've purchased either the *TeXbook* or the LaTeX *Guide*. (Both from Addison-Wesley. You can find these at any university bookstore and at some computer stores. Radical Eye Software doesn't include them with the software because it would increase the price of AmigaTeX dramatically and many users likely own them already.)

Any text editor can be used to create the TeX source files. I would recommend a customizable editor such as *DME*, *UEdit*, *TxEd*, or Tom Rokicki's *mg* editor, available separately from AmigaTeX. (AmigaTeX doesn't include an editor.) The output is a DVI file which can then be printed with one of the printer drivers or previewed onscreen with the AmigaTeX *Previewer*.

AmigaTeX is a strictly CLI-oriented system, since disk space is at a premium. The manual provides a basic introduction to the CLI for those users who might usually use the Workbench instead.

AmigaTeX also provides support for the ARexx scripting language.

The *Previewer*

The AmigaTeX *Previewer* is the gem of the whole system. While the TeX software is merely a port (but still a lot of work!) of CTeX, a version of TeX written in C by Rokicki, the *Previewer* is original Amiga software written from scratch. The function of the *Previewer* is simple: allow the Amiga user to view TeX output on the Amiga screen with as much detail and accuracy as possible. It succeeds wonderfully.

The *Previewer* displays the TeX DVI files using the Amiga's hi-res graphics. It supports both the interlaced and non-interlaced modes and popular utilities such as the *morerows* program. The complete set of TeX fonts can be displayed by the *Previewer* at various magnification levels, as can the (limited) TeX-generated graphics. Fonts (about 1500 of them) are loaded from disk as they are needed by the *Previewer* - a font cache helps speed things up for commonly-used font types and sizes.

The *Previewer* can track a TeX file as it is being processed and display the resulting output onscreen as soon as a page is finished.

Mathematical formulas are easy in TeX: saying

```
\[L=\int_{a}^{b}\sqrt{1+\left(\frac{dy}{dx}\right)^2}\;dx\]
```

results in

$$L = \int_a^b \sqrt{1 + \left(\frac{dy}{dx}\right)^2}\, dx$$

as output.

Tables are also easy:

Year	World Population
8000 B.C.	5,000,000
50 A.D.	200,000,000
1650 A.D.	500,000,000
1850 A.D.	1,000,000,000
1945 A.D.	2,300,000,000
1988 A.D.	5,000,000,000

Printer Drivers

A printer driver that supports your printer must also be purchased to print the DVI files created by AmigaTEX. Printer drivers come in sets for 9-pin, 24-pin and laser printers, ranging anywhere from 2 to 10 diskettes. Support is available from Radical Eye Software in creating drivers for non-standard printers.

If you want professional results, a laser printer is your best bet; but if you only have access to dot-matrix printers, a 24-pin printer is preferable to a 9-pin printer. The *Previewer* should be used whenever possible to reduce the editing time for the document, reserving hard copy only for the final editing stage.

MetaFont

Rokicki has also ported Knuth's *MetaFont*, used to create and modify TEX fonts. *AmigaMF*, as the version is called, is available separately from Radical Eye Software.

Documentation

A new 186-page manual is included with AmigaTEX. The manual gives you all the information necessary to run AmigaTEX on your Amiga, some history about TEX, some specialized technical information and some sample font outputs. The manual is well-written and includes sections geared to the newcomer to the Amiga.

Support

AmigaTEX is actively supported by Tom Rokicki, who is more than willing to help a user with any problems. A new BBS dedicated to AmigaTEX should be in operation by the time you read this.

Conclusions

If this review sounds laudatory, that's because it is. I don't buy much software for my Amiga, but I certainly have no regrets about purchasing AmigaTEX. It works perfectly - the *Previewer* especially is fantastic. The portability aspect is a real bonus. I regularly copy my DVI files onto an IBM 3.5 inch diskette and print them on an Apple LaserWriter using a TEX implementation on an IBM PS/2. Try that with your word processor...

Again, a cautionary note: AmigaTEX is a superb TEX implementation, but TEX may not be what you want. It's not cheap once you throw in a printer driver, the *TEXbook* and the LATEX *Guide*. I would check the latter two first to see if TEX is what you're looking for. And if you are interested, a demo version of AmigaTEX is available free from Radical Eye Software (details noted above).

A final note for die-hard TEX users: if you want TEX on your Amiga, AmigaTEX is one of the best implementations around on any computer.

Transactor for the Amiga is now being published monthly!

Get twelve great issues every year instead of six

See order card for subscription information

TransAmi Article Index

Volume 1, Issue 1

Thoughts On An Amiga Programming Environment *by Perry Kivolowitz*
The Packet Interface to AmigaDOS in C *by Matt Dillon*
The SCA Amiga Virus *by J.C. Bat*
BreakPoint *by Victor Wagner*
$13 Common Amiga Programmer's Errors *by Bryce Nesbitt*
Porting from Unix to Amiga *by John Toebes*
An Introduction to Memory Allocation and Lists *by Rob Peck*
In the Beginning was CAOS *by Andy Finkel*
A Programmer's Tour of "arp.library" *by Scott Ballantyne*
Amiga Machine Language - A Detailed Example *by Jim Butterfield*
...And What Are Little Fonts Made Of? *by Betty Clay*
Warrior Cycles *by Rico Mariani*

Reviews: Lattice 4.0, Power Windows 2.0, saf-T-net, Byte Box A500 RAM expansion

Volume 1, Issue 2

BreakPoint, Part 2 *by Victor Wagner*
Preview of AmigaDOS V1.3
AmigaDOS V1.3 Manual Update
Workbench Musings *by Rob Peck*
Info about .Info *by Betty Clay*
Random, or Just Confused? *by David Wood*
About Draco *by Chris Gray*
Draco - the Language and Compiler *by Chris Gray*
Representing Objects in a CAD System *by Charles B. Blish*
Programming With AmigaDOS *by Bob Rakosky*

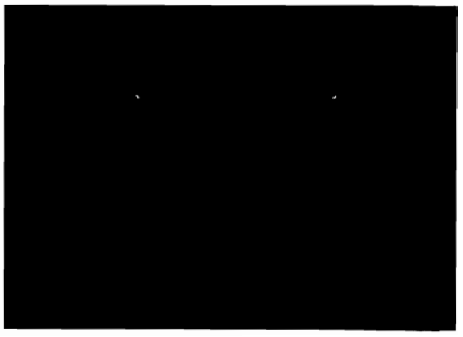
Reviews: AssemPro, Photon Paint

Volume 1, Issue 3

BreakPoint, Part 3 *by Victor Wagner*
3-D Plotting for AmigaBasic *by Anthony Bryant*
Ray Tracing Fundamentals *by Cathryn Graham*
Turbo Tracing *by Louis Markoya*
LUCAS - A 68020/68881 board for the A1000 *by Brad Fowles*
The Final ARP.library Tour *by Scott Ballantyne and Charlie Heath*
The ANSI Standard for C *by Eric Giguère*
Internationalizing Your Programs *by John Toebes*
Survival Guide to the BridgeBoard *by Dan Schein*

Reviews: Manx Aztec C 3.6 and sdb, ARexx

Volume 1, Issue 4



The Changing of the Fonts *by Betty Clay*
Calling the CLI from AmigaBasic *by Jim Butterfield*
The Future of Amiga Computing *by David Wood*
Creating Run-Time Libraries *by Steve Simpson*
Amiga Modula-2 Programming *by Clark Williams*
Programming in M2Amiga *by Anthony Bryant*
Reading C Declarations *by Eric Giguère*
Some examples of shell programming *by John Faichney*

Review: WordPerfect Library

Book Review

The C Programming Language, 2nd Edition

by Eric Giguère

Eric Giguère has been involved with the design of an ANSI-conformant C compiler at the Computing Systems Group at the University of Waterloo, where he is a computer science student in the co-op program.

The book *The C Programming Language* has been a veritable cash cow for both its authors and its publisher since recent surge in popularity of the language in question. In the world of computer publications, it's a classic, the C programmer's bible. So what would you do with such a book? Bring out a second edition, of course.

In all fairness, though, the appearance of a new edition is more than justifiable. The first edition was published ten years ago, an unbelievably long time in the fast-moving field of computers. No one back then could have forseen the popularity of the language today. As well, a formal, standard definition of the language is near final approval by the American National Standards Institute (ANSI), as was detailed last issue in this magazine. The first edition is thus hopelessly outdated, a situation the authors have somewhat corrected with the publication of the second edition.

A few words on nomenclature: throughout this article, the two editions of The C Programming Language will be referred to as "K&R", a common abbreviation due to the authors of the book, Brian Kernighan and Dennis Ritchie. As well, the draft proposed ANSI standard for C will simply be referred to as "the Standard".

First Impressions

At first glance, the second edition of K&R seems quite similar to the first. The cover is a bit fancier - three colours instead of one - but still has the big blue "C" on a white background. Both editions are of similar size, the second edition being about 50 pages longer. The table of contents are almost identical. And both are set in the same type. So is there a difference? Well, yes and no.

Tutorial

The eight chapters that make up the main text of the book serve as a tutorial introduction to programming in C. As far as I've been able to tell, the text and examples are nearly identical to those found in the first edition, though parts of certain chapters have been completely rewritten to reflect significant changes made by the Standard. Many of the programming examples have been altered to reflect the Standard as well, and there are more pictorial examples, but overall it's the same.

This isn't a criticism, though. K&R is a well-written introduction to programming in C for users with some programming experience. There are enough examples covering a wide variety of topics that even experienced C users will find themselves thumbing through the book in search of enlightenment.

One thing I like about the second edition is that references to the UNIX operating system aren't as widespread and are concentrated mainly in the chapters dealing with file input and output. In the early days, C and UNIX were so intertwined that many programmers had difficulty keeping them apart. Since then, however, and especially with the introduction of the Standard, C has migrated to many different operating systems. As such the second edition of K&R presents a much more generic approach to the language.

Reference

The three appendices - two more than in the first edition - take up approximately seventy pages and are likely to be the parts of most interest to experienced C programmers. Appendix A is

"In the early days, C and UNIX were so intertwined that programmers had difficulty keeping them apart... the second edition of K&R presents a much more generic approach to the language..."

a complete rewrite of the C Reference Manual found in the first edition; Appendix B summarizes the library routines defined in the Standard and their associated header files; and Appendix C gives a quick summary of the major changes between the Standard and the first edition of K&R.

The new C reference manual is a much more precise definition of the language than the one found in the first edition. It is a complete summary of the Standard, meant for the programmer rather than the compiler implementor (this is a blessing because you wouldn't want to wade through the actual Standard itself - trust me). A grammar for the language suitable for LALR parsing (after modification) is also included.

Appendix B, the Standard Library summary, will be of particular use to those interested in writing portable programs. The functions described in this section must be supported by any ANSI-conformant compiler. What's missing, however, are the exact contents of a few more header files. These are just as useful as the function prototypes themselves.

Appendix C is only three pages long and makes for very light reading. If you read the ANSI summary last issue, you know more than what's presented here.

Final Analysis

To me the timing of the second edition is all wrong. They should have either published it three or four years ago, when C was undergoing its first big surge in popularity, or waited until the final version of the Standard was approved by ANSI. Already the current draft departs from what K&R says (I'll be presenting an update on the Standard in a later issue of Trans-Ami). So I wouldn't rely on it as the final arbitrator as to what is "legal" C.

But as a general reference book it certainly is one of the best. I would certainly recommend it to newcomers to C - along with the appropriate manuals for their specific machines and compilers. Should you go out and buy it if you already have the first edition? That's a tough question. I would recommend that you check it out in a local bookstore first. Flip through it. Do you really need what's new? If you're just looking for a summary of the proposed Standard, simply refer to the last issue of this magazine. And until the Standard is formally approved, nothing is final.

Let's see, I've got forty bucks to spend... what am I going to do? Get the new C book? Or get that great new arcade game for my Amiga? Hmmm...

The fastest
Modula-2
Software Development System for
AMIGA
$ 249
Demonstration disk $ 10

Modula-2 is the successor of Pascal. Both languages were invented by Professor Niklaus Wirth. Modula-2 is an all purpose language suited for educational, home and professional use. Modula-2 leads to good programming technique, easy readability and maintainable programs. In fact, a lot of schools and universities have chosen Modula-2 for their programming courses.
M2Amiga is a software development system for Modula-2 based on an extremly fast single pass compiler. It is fully integrated into Amiga Workbench and is easy to use from CLI or Workbench. It runs on all Amiga models with a minimum configuration of 512k RAM and one disk drive.
M2Amiga has been developed specially for the Amiga and therefore optimally supports the features of this unique computer.
- Single pass compiler produces optimized native MC68000 code from standard ASCII files. Full support of latest edition of Modula-2.
- Fastest library calls possible on the Amiga. No intermediate code needed for library calls and parameter passing.
- Supports FFP, 32 bit IEEE and 64 bit IEEE reals formats, you can even mix all three formats within the same program.
- Easy access to MC68000 registers, allows the insertion of inline code.
- Full interface definition to the Amiga's operating system following the official definitions.
- The best runtime system currently available on the Amiga. Opens and closes all used libraries, captures all GURUs and releases all used resources after program crash or termination. Routines for System requesters enable user interface consistent with the Amiga's Intuition.
- The emacs based editor places the cursor on compilation errors and explains them in full text.
- In just a few seconds the linker produces stand-alone applications that can be run from CLI or Workbench. Stand-alone applications do not contain any useless overhead – they are compact and fast.
- All programs can produce icons for their output files to allow effective use of the intuitive Workbench environment. Experience the new feeling of program development by selecting and double clicking icons!
- Also included are a project management tool and other utilities.
- Besides the handy manual, an introductory guide to Modula-2 is included. This documentation allows both the beginner and the advanced user to find any information needed quickly and easily.
- Demo programs show the use of various features of the Amiga.
- For professional program development special utilities are available.
  + Source Level Debugger, the new way of debugging your programs. Inspect all your variables (displayed in their own format), zoom into structures as records or arrays or follow pointer chains by clicking the mouse buttons
  + Object file converter, converts standard Amiga object files into M2Amiga format. This allows linking of routines written in other languages.
  + Linker for devices or libraries. Add power to the Amiga by supplying new libraries and devices. Easy and intuitive programming instead of fooling around with Assembler and Amiga linker are possible now!
  + Modula-2 Amiga Programming System Environment

We have Modula-2 Compilers for HP-UX, IBM/370, PCs (Taylor, M2SDS, JPI), OS-9 and Sun, tomorrow even more!
Texas residents add 8% sales tax. International Orders add $ 20 shipping.

The Modula-2 people:
INTERFACE TECHNOLOGIES
3336 Richmond, Suite 323, Houston, TX 77098-0990 (713) 523 8422
Dealer Inquiries welcome

A. + L. Meier-Vogt
Im Späten 23
CH-8906 Bonstetten/ZH
Switzerland
Tel. (41) (1) 700 30 37

Book Review

Amiga Tricks & Tips

by Jim Butterfield
Copyright © 1988 Jim Butterfield

Amiga Tricks & Tips
By Bleek, Maelger and Weltner
Published by Abacus Software
(A Data Becker Book)

I have mixed feelings about this book. It's got the wrong title; it might well have been called, "Digging into the Amiga". It contains very good stuff indeed, mixed in with annoying trash. It shows excellent programming techniques, but seldom explains them. You need to wade through the bad to get to the good... but the good stuff is there, and it's very valuable indeed.

A disk is available to go with the book, and I suggest you get it; it will save you a great deal of typing, not to mention eliminating keyboard errors. The book contains a considerable number of programs, many of which you will want to build on for your own programming work.

I wish the book started at page 37, since that's where the meaty material begins. Before that, there's a collection of material on the CLI and startup-sequences, much of which is outdated and some of which is wrong. I was particularly annoyed by the discussion of templates on page 12; their explanation of /A, /K, and /S is quite misleading, and I was tempted to close the book at that point and give up on it. I'm glad I didn't; when the book hits its stride, it is very good indeed and handles material that is hard to find elsewhere.

This book does not deal with C, Assembler, or other non-Basic languages, but it does something that will be of value to anyone planning to program the Amiga in any of these languages. It gives you some first steps in tackling the Amiga system libraries in order to exploit the machine's power. This is often where beginners get blocked; their language textbooks usually don't say a thing about how to do the custom coding that makes the Amiga sparkle.

You'll get this here, positioned from a Basic point of view. Want to set up a borderless window, create a mouse-driven "rubber band", rummage through all files in a drawer, or handle a disk sector directly? It's all there and more, embedded in illustrative programs. Explanations are often thin, but the code speaks volumes. I wish that the various system libraries - DOS, Exec, Intuition, Graphics and others - were more specifically identified as they were tapped. You may want to keep other reference guides on hand to help sort it out. But the programs work; from there, you will have a base from which to develop your own techniques.

If the book's objectives had been better defined, it would probably have spent more time tracking structures in addition to its excellent job of exploring system libraries. It seems like a shame to see programs that assume the use of the 80-column character set (topaz 8) when it would have been so easy to expend a little extra effort to find the size of characters actually being used. It's not hard to track from the Window structure, perhaps to the RastPort, and get this information; but the book doesn't go that way. Similarly, the chapter on error trapping starts by suggesting that standard AmigaBasic has a hard time telling whether a file exists on a diskette or not; in fact, a simple error trap will do the job. Amazingly, the standard error trap, ON ERROR, is never discussed in this chapter!

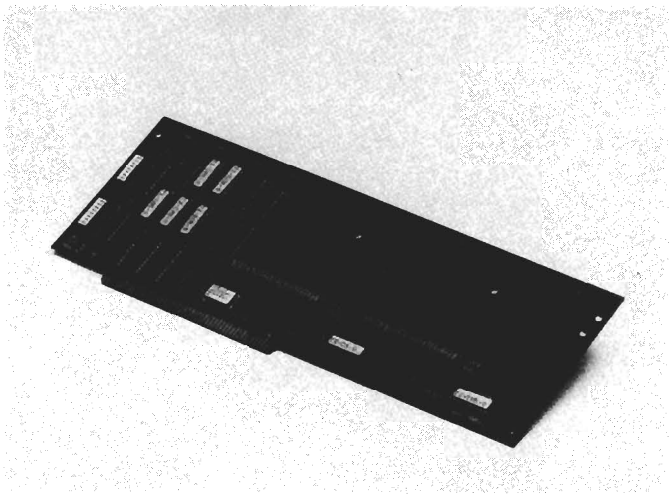
I suppose I'm easily irritated by small things. When Chapter 2 starts by saying, "The newest version of the Workbench is here!", I am irked to discover, a couple of paragraphs further along, that the book is talking about 1.2. I understand how it happens, but I wish they had identified the version in the opening sentence. "Effective Programming" is a nice section title, but contains little style advice: instead, it contains Basic benchmark programs of limited usefulness and a piece on creating shorter .bmap files. Gosh, with AmigaBasic you can develop a splendid programming style... but this isn't the place to look for it.

But let's set aside the difficulties and disappointments. The book does something that's hard to find: it lets the programmer boldly go where few programmers have gone before. Deeper inside the Amiga.

If you want to know your machine better - if you want to smooth the entry path for other languages - this book is valuable. Despite its shortcomings, it gives you material that's hard to find elsewhere. It's a good launching pad to the inner space of your Amiga.

News LINK

Twin-X IEEE 959 interface from ASDG: Twin-X is an Amiga 2000 host for a widely used standard expansion interface called IEEE 959. This interface (also called iSBX™ by Intel) was originally developed to allow mini and microcomputers to perform process control and data acquisition, and IEEE 959 modules performing an extremely wide range of functions have since been developed. According to ASDG, there are literally hundreds of modules available from approximately 30 vendors, including controllers for GPIB (IEEE-488), disk and tape drives, video equipment, stepper motors, optical scanners, A/D converters, digital speech modules, Ethernet, I/O ports, floating point units, etc. The Twin-X is a standard A2000 plug-in card that can host one or two IEEE 959 modules, and allows modules to auto-config under Kickstart 1.2 or later. Contact Perry Kivolowitz, ASDG Inc., 925 Stewart St., Madison, WI 53713. Phone (608) 273-6585.



ASDG SpectraScan Amiga scanner: ASDG announced in October that shipments of their professional colour image input system, SpectraScan, would begin at the end of the month. SpectraScan is based on the Sharp JX-450 colour scanner and ASDG's new Twin-X general purpose I/O board, and provides high-resolution full colour image capture to the Amiga 2000. The Sharp JX-450 is a flatbed scanner accomodating colour originals up to 11 by 17 inches. With an optional mirror attachment, it will also scan colour slides, negatives or transparencies. Horizontal and vertical resolution can be set independently from 30 to 300 dots per inch to produce pixels of any aspect ratio. Twenty-four bits of colour information is reported per pixel, resulting in a palette of 16 million colours.

Operation of the scanner is completely controlled by ASDG's SpectraScan software, which has an "image intensive" user interface, displaying the scanned image on the screen at all times. To define a sub-range of the image to scan, the user sweeps a rectangle over the displayed image, which scrolls smoothly if the area to be enclosed goes beyond the edge of the display. Another feature of SpectraScan is its conversion from the scanner's 24-bit pixel data to the Amiga's graphics modes, including HAM mode. A unique capability of SpectraScan's colour mapping is that it is not limited to powers of two; it will convert an image to exactly 59 colours if that's what is needed. This is useful when one or more dedicated colour registers are needed in an image, for example in video titling. Since the raw data for a scanned image can be very large (52 megabytes for a full page, full colour scan), SpectraScan uses a demand-paged virtual memory system to allow manipulation of these images on a 2 megabyte Amiga with a hard drive. A full-size 11 by 17 inch image, creates a HAM image 3300 pixels wide by 5100 pixels high, which will be smoothly scrolled on the screen. Furthermore, SpectraScan never uses more than 200K of chip memory, allowing you to run other applications at the same time. A direct interface with desktop publishing programs is possible, since other programs can operate SpectraScan directly, and read the files it produces. SpectraScan includes a Twin-X, SBX-GPIB module, all cables, SpectraScan and GPIB software with manuals, and retails for $995 US. ASDG also distributes the Sharp JX-450 scanner; the retail price is $6995 US. For more information, contact Perry Kivolowitz, ASDG Inc., 925 Stewart St., Madison, WI 53713. Phone (608) 273-6585.

Gold Disk offerings: MovieSetter is a WYSIWYG animation program. The carefully-designed, simple to use interface amkes MovieSetter an Amiga animation package of unprecedented speed. MovieSetter allows you to rapidly produce complex animation sequences of several minutes duration with only 1MB of memory. Use built-in, professionally drawn movie-clip files, or create your own 32 colour images within the program. MovieSetter will also import IFF graphics from popular Amiga programs including Deluxe Paint II and ComicSetter. Simply use the mouse to change positions of characters and objects over static or scrolling backgrounds. Generate sounds within the program altering duration, pitch and volume. Even attach stereo sound to characters and listen to them move from on side of your monitor to the other. Special features such as colour cycling, playback at up to 60 frames per second and linear and elliptical guides help you create special animation effects. And full video overscan allows you to save feature length productions to your home video library. List price: $99.95 (US) or $124.95 (CDN).

Professional Draw is a high-powered graphic arts tool that Gold Disk says "will become a standard design instrument for any Amiga user - from the novice artist, to architects and planners, to the most discriminating advertising professional. Professional Draw is the ultimate colour illustration and graphic design solution." Professional Draw offers a varied assortment of powerful drawing tools including circles, ellipses, rectan-

gles, polygons and bezier curves. Import existing IFF files or HAM images and transform them into fully editable structured drawings using 4-point continuous beziers, user-definable line weights and fill patterns, and a virtually unlimited range of colour. Or use built-in clip-art and text character macros. Sophisticated transformation features including mouse-controlled rotation, scaling, shearing and distortion of images provide for high-calibre results. The macro features allow recall of a previously designed image at a single keystroke. Output detailed graphics to the highest resolution of dot matrix devices including: 24-pin printers, colour printers and inkjets, and HP LaserJets and compatibles. Or output to any PostScript device including high-resolution typesetting equipment for professional results. Professional Draw's built-in colour separator will prepare your colour graphics for mechanical or process colour offset printing. Suggested retail is $199.95 (US).

Design-3D is a modelling package which allows the user to model in wire frames with 6 line types and 16 colours (with dithering to emulate additional colours), and fast flat shading. Design-3D provides 4 views - top, side, front, and perspective with 4 light sources. Axial extrusion allows for very fast modelling of symmetrical objects. Objects may be imported from VideoScape 3D. Support for 3D text (fonts included) adds a creative edge to Design-3D. The program's built-in font editor provides tremendous control over this feature. Design-3D allows you to work in high-resolution, interlace or non-interlace, in any unit of measure - inches, centimetre, etc. The ANIM format is supported to import images into PageFlipper Plus or animate within the program in real time. Requires 1MB. List price: $99.95 (US), $124.95 (CDN).

Gold Disk has also developed a hardware/software interface for the Canon IX12, black and white, 300 dpi scanner. Ideal for Amiga desktop publishing or illustration, this scanner will allow the user to save images as black and white IFF files at 75, 150, 200, or 300 dots per inch. Simply feed your page into the scanner. The image will be saved in approximately 15 seconds. Software control allows the user to define regions of scanned pictures directly into your Professional Page layouts. You can even import these scanned images into Professional Draw to be traced over and made into structured graphics. Comes with parallel port hardware and software interface. Suggested retail price: $1095.00 (US). For more information about ComicSetter, MovieSetter, Professional Draw Design-3D, or the Canon IX12 scanner, contact: Gold Disk Inc., P.O. Box 789, Streetsville, Mississauga, ON, Canada, L5M 2C2, (416) 828-0913, FAX (416) 828-7754.

FlickerFixer Awarded FCC Class B Certification; European Amiga 2000 Compatibility announced: MicroWay announced that flickerFixer, its advanced graphics adapter for the Amiga 2000, has been awarded Class B certification by the FCC. With the flickerFixer graphics card, the interlace flicker and visible scan lines of the Amiga 2000 display are completely eliminated. FCC Class B status permits the flickerFixer to be used in a home as well as business or educational environment. FlickerFixer is now PAL compatible with European versions of the Commodore A2000. As a result, European Amiga users can display video output (up to 704 x 510 pixels) flicker-free on high resolution and VGA monitors. FlickerFixer is fully compatible and transparent to all user software. The price of the flickerFixer is $595. For further information, contact Steven Bennett, Amiga product manager, (508) 746-7341.

Excellence! from Micro-Systems Software is a full-featured WYSIWYG word processor for the Amiga. Its features include: full colour support of text, inclusion of IFF graphic images, spelling check as you type, math capabilities within documents, multiple column support, proportional font support, index generator, table of contents generator, integrated thesaurus, integrated grammatical and style checker and PostScript output. Excellence! shows everything on-screen, including super and subscripts, headers, footers, footnotes, colours and graphics. From Micro-Systems Software, 12798 West Forest Hill Blvd., Suite 202, West Palm Beach, FL 33414. For inquiries, call 1-800-327-8724. In Florida, (407) 790-0770.

The Disk Mechanic is a comprehensive collection of AmigaDOS utilities for every application. The Disk Mechanic can recover files that have been deleted, salvage files from corrupted disks, and repair damaged files. It includes a disk optimizing program that can increase your hard or floppy disk access speed up to 400% by reorganizing the disk's data. Also included is a hard disk backup program and a full-featured disk block editor for the advanced user. Version 2.0 of The Disk Mechanic includes full support for the new Amiga fast filing system and a new high-speed hard disk backup utility. From Lake Forest Logic Inc., 28101E Ballard Road, Lake Forest, IL 60045. Phone (312) 816-6666

Neriki Image Master Pro-Genlock is a full broadcast quality genlock designed specifically for the Amiga. It includes a 520 line NTSC (600 line PAL) encoder, and the chroma burst level meets all broadcast standards. The Neriki Image Master Pro-Genlock accurately genlocks the Amiga's RGB output to a reference video signal or a studio black signal and keys the computer image into the reference video by using a fully adjustable luminance keyer. With this genlock, the Amiga becomes a highly professional tool perfect for video titling, animation and graphics. The Neriki fully supports all graphics modes and resolutions and is compatible with all Amiga models. The Neriki encodes the Amiga's RGB output into a broadcast quality composite video signal without degrading the quality of the reference video, thus minimizing duplication losses. Edited second-generation masters overlaid with Amiga graphics and titles can be produced using the Neriki Image Master, with third-generation production tapes retaining excellent quality. The unit incorporates both a fully-controllable luminance keyer and a key to background colour function. Distributed in Canada by Compu-Art, P.O. Box 712, Victoria Station, Montreal, PQ, H3Z 2V8. Phone (514) 483-2080.

Microfiche Filer Plus: Software Visions, Inc., the manufacturer of Microfiche Filer, announced and demonstrated its new professional database, Microfiche Filer Plus, at AmiEXPO in

Chicago. Microfiche Filer Plus provides all of the speed, power and ease of use of Microfiche Filer, as well as many new and advanced features. Among the many new features are: automatic field calculations, HAM and overscan graphics support, more sophisticated printing features, and optional macro programming using the ARexx macro programming language. Using ARexx macros, you can automate many database operations and manipulate data using the full power of a programming language. Using ARexx's interprocess communication facilities, you can import and export data in any format and pass data directly between Microfiche Filer Plus and other ARexx-compatible applications. The suggested list price of Microfiche Filer Plus is $179 (US); the original Microfiche Filer remains at $99. Both products are available from Amiga dealers and distributors worldwide. For more information, contact: Software Visions, Inc., P.O. Box 3319, Framingham, MA 01701. Phone 1-800-527-7014.

Four Megabyte RAM Expansion: Digitronics is now shipping its RC4 Ramcard for both the Amiga 500 and Amiga 2000 computers. The RC4 is a 100-pin expansion card that plugs into one of the A2000's expansion slots. It is socketed to accept 32 standard sized DIP one-megabit memory chips, and can be configured as low as 1/2 a megabyte. Memory can be added in 1/2 megabyte increments as desired, up to four megabytes. Given the unstable pricing of RAM chips lately, this allows the user to start out inexpensively and add gradually as their budget allows, while not limiting their expansion to only two megabytes. The RC4 uses custom refresh circuitry to reduce power consumption and to eliminate CPU and refresh contention, resulting in the fastest memory board possible for the Amiga. For Amiga 500 owners, Digitronics offers an expansion box which slides onto the expansion connector on the side of their machine. The RC4 memory card fits inside this case. The expansion box is only one inch high, and fits neatly underneath an external disk drive. It comes with its own power supply, and powers itself on and off with the A500. The RC4 comes with diagnostic software that will test each RAM chip individually, as well as software that will enable the RAM to be used as a recoverable RAM disk. Suggested retail prices are $225 for the RC4 Ramcard only (0k memory), and $325 for the RC4 and expansion box. The RC4 is also available in incomplete kit form for $100 (all prices in US dollars). For more information, contact: Digitronics, P.O. Box 579, Hatfield, PA 19440. Phone (215) 361-1999.

The GOMF Button: Hypertek/Silicon Springs announced GOMF 3.0 - "The GOMF Button" (hardware/software), and GOMF 3.0 (without hardware). The software is an upgraded version of the popular "GOMF" software package and is compatible with 68000 and 68010 microprocessors. In addition, the C00000 memory problem with Amiga 2000 computers has been fixed with the new version. The GOMF Button offers the user the ability to actually save his data after a program has crashed. In addition, the most catastrophic error - the 'freeze' - can be eliminated. With a simple push of the button, locked up machines unlock, and your data is preserved. The GOMF Button installs in any Amiga in minutes, and its circuit board is placed under the PAULA chip. From there, a small wire with a pushbutton protrudes out the case. Alternately, a hole for the button can be drilled in the case. Suggested retail price is $69.95 for The GOMF Button, and $34.95 for the software only. Registered owners of the previous versions may upgrade to The GOMF Button for $30. The software upgrade is $10 (prices in US dollars). Both upgrades include a new manual, bound in a sturdy vinyl binder. These upgrades will be handled directly from Hypertek/Silicon Springs. For more information, contact: Dean Spyropoulos at Hypertek/Silicon Springs, (604) 942-4577.

GFA BASIC 3.0: This BASIC provides an interactive editor that is able to identify the most common programming errors before the program is run. It is compatible with the Atari ST version, so that the many GFA-BASIC books and program listings already available can now be used on the Amiga. Special features of the Amiga version: • Designed for the 68000 and 68020 processor • reentrant: GFA BASIC does not need to be reloaded for each additional task • The graphics support is powerful and easy to use: for example, you can create a filled polygon using only the command 'polyfill', saving dozens of operating system commands • Menu lists and pull-down menus are created with the Menu Construction Tool and special GFA-BASIC commands. • All operating system functions are available through GFA-BASIC 3.0, most as GFA-BASIC commands. Suggested retail price is $99.95 (US). From Michtron, 576 S. Telegraph, Pontiac, MI 48053. Orders and information: (313) 334-5700.

City Desk 2.0 Desktop Publishing Software has many new features that enhance the power and flexibility of City Desk 1.1, including: Text flow around irregular graphics, auto hyphenation, full-screen editor, new graphics editor, and rulers and *x,y* coordinate display. Version 2.0 has increased PostScript and HP series II font support and new gray scale support with four shades on-screen and up to 16 on the printed page. Suggested retail price is $199.95 US. Registered users of City Desk 1.0 and 1.1 may upgrade to Version 2.0 for $35.00. Contact Microsearch Technical Support Group for details at (713) 988-2818.

EyeBytes Computer Visuals: Based in Washington state, Eyebytes offers solutions to those using Amigas for cable broadcasting and want to use the machine to its full potential. For cable companies not using Amigas but wanting to update beyond a dry scrolling text format, EyeBytes offers extensive services ranging from purchasing equipment and selection of software to training the staff producing a channel. EyeBytes also have the experience to train users in the fastest-growing fields like desktop video production and point of sale advertising. Unique is the best way to describe any system EyeBytes places or updates. For further information contact: EyeBytes Computer Visuals, Deborah Devine, CEO, 1900 Brooklane #E9, Ellensburg, WA 98926. Phone (509) 962-6570.

Engineering hardware and software from ACDA: ACDA Corporation, developers of medical systems and data acquisi-

tion and process control devices for scientific and enginerring applications, now brings its technology to the Amiga. By enhancing the Amiga with ACDA's family of hardware and software products, technical users can configure the Amiga as a true scientific and engineering workstation. Data acquisition and process control products include: Proto 40K, a versatile, multi-channel A/D and D/A digital I/O card ($1895); Proto 5K, a single channel A/D I/O device ($279.95); Amiga GPIB, an IEEE-488 controller card ($395); and Digiscope, an oscilloscope emulator package ($139.95). Programming tools include: AmigaFFT, a package of Fast Fourier Transform routines ($152); and AmigaView, an object-oriented C language interface with over 100 routines and macros designed to reduce programming time and code for professional applications development ($79.95). Prices in US dollars. ACDA Corporation, 220 Belle Meade Ave., Setauket, NY 11733. Phone (516) 689-7722.

Special Effects Stereo Sampler: Starvision International has introduced The Omega Stereo Sampler, a professional quality stereo sampling digitizer for the Amiga. Developed in Denmark, The Omega Stereo Sampler offers high quality stereo sound reproduction that eliminates unwanted noise while being the smallest sampler on the market today. The Omega gives real-time stereo sampling, with dual channel input controls, without necessary amplifier connection. It comes complete with sampling software and a comprehensive manual that describes the best sampling methods and techniques. A microphone accessory is also available. Suggested retail price is $125 US. For more information or to order, call (212) 867-4486. Starvision International, 305 Madison Avenue, Suite 411, New York, NY 10165.

The ToolCaddy Docs: Online help for the Amiga programmer, in three versions: 1. the entire complement of Amiga function calls; 2. each of the Amiga structures contained in the "include" files; 3. all the MC68xxx instructions. The three versions work together by sharing internal resources, but may be used individually as stand-alone programming aids. The ToolCaddy Docs remain in the background and pop up when needed, letting you view the reference screens, or the more detailed examples by making selections from an index. The text files (including sample source code) may be "cut" and saved, or printed. The system is disk-based, so total RAM requirements never exceed 12K. Suggested retail is $29.95 for each version. Available at dealers or order direct from: The ToolCaddy Works, P.O. Box 5873, Laughlin, NV 89029-5873. (No credit cards or C.O.D.)

Home Remote Control with the Amiga: From Compplications comes *PowerMaster*, a program for use with the X-10 Home Computer Interface. Lights and appliances are plugged into X-10 modules and inserted in a wall socket; *PowerMaster* uses menus, requesters and gadgets to define module names and locations, and to program up to 128 ON, OFF and DIM times for up to 256 modules. Multiple definitions for different times of the year may be stored in "event" files. Event definitions are sent to the X-10 interface through the serial port; after programming, the battery-backed X-10 interface no longer needs the Amiga, and it can be removed. The X-10 interface then sends commands to the remote modules over existing house wiring at the times programmed by *PowerMaster*. X-10 modules are widely available under various brand names such as Sears, Leviton, Stanley, GE, Radio Shack and others. *PowerMaster* is sold either by itself ($22 + $2 s/h), or packaged with an X-10 computer interface ($48 + $3 s/h). For ordering (specify A500/1000/2000) or more information, contact: Compplications, 1727 Parkview, Redlands, CA 92374. Phone (714) 794-5311.

Professional CAD Software for the Amiga: Haitex Resources recently released X-CAD Designer in North America, a general-purpose 2D design tool incorporating features for both architectural and mechanical design users. According to Haitex Resources' president Shawn Glisson, "the graphics power of the Amiga has given X-CAD an incredible advantage in price/performance over IBM-PC based systems". X-CAD displays over 1,100 lines per second, putting it in the supermicro-workstation bracket. A recent benchmark test by CAD/CAM International serves as an example: on an IBM PC/AT, AutoCAD took 2 minutes to display 1,000 circles and Robo-CAD took 1.5 minutes. By comparison, X-CAD displayed the circles in only 10 seconds on an entry-level Amiga system. The addition of a 68881 math co-processor even further increases X-CAD's advantage. X-CAD Designer retails for $599; a complete hardware/software X-CAD system is less than the price of competing PC software alone. Distributed in the U.S. by Haitex Resources, 208 Carrolton Park, Suite 1207, Carrolton, TX 75006. Phone (214) 241-8030.

C Programmers Association: Tired of re-inventing the wheel? Thinking that someone, somewhere must already have a bug-free version? Stuck at two A.M.? Need only that one little routine to make it work? "C-Pro" brings you: broad access to C functions - choose from hundreds of general and machine-specific C functions, and receive them instantly through C-PRO's unique BBS system; High standards - all C functions offered by C-PRO are subject to stringent evaluation for usability, efficiency and bug-free operation; Profit - upon favourable review, C-PRO will distribute your C functions and give you a healthy percentage of the profit for each one sold; Confidentiality - mutual non-disclosure agreements ensure that any programs you submit will not be used without your approval. For more information, contact: C Programmers Association, 10668 Ellen Street, Suite D, El Monte, CA 91731. Phone (818) 442-1522 and ask for Walt Soden.

Lattice brings C++ to Amiga: The Lattice C++ is a preprocessor that translates the C++ language into C source code which is then compiled by the Lattice C compiler and linked to create an executable program. C++ is a superset of the C language, allowing greater data abstraction and object-oriented programming. Because C++ is object oriented, it is particularly well suited for the Amiga where the operating system is defined in terms of objects. By supporting a feature known as "inheritance", C++ allows simple objects to be expanded further.

The Moniterm Viking 1 Monitor for A2000: The Viking 1 for the Commodore Amiga 2000 targets business and professional users for desktop publishing, CAD/CAM engineering workstations, presentation, and spreadsheet applications. A2000 users can benefit from a large-screen, non-interlaced high resolution monitor for these demanding tasks. The Viking 1 19-inch paper-white screen provides four intensity levels for gray-scale use, combined with 1008 by 800 pixels for high resolution. The display will support previous lower resolution modes (including 320 by 200 and 640 by 400) with its non-interlaced design at a 63 Hz refresh rate for clear, crisp images.

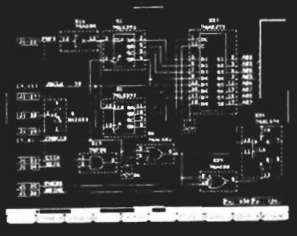
The Viking 1's rock-steady, flicker-free display shows two full facing pages of text and graphics for desktop publishing and word processing, a B-size engineering drawing for architectural and CAD/CAM applications, and a 13-month spreadsheet with 68 rows and 138 columns. Common screen disturbances such as smearing, edge distortion, defocusing and flicker are eliminated with the Moniterm Viking "ergonomically correct" design. For more information, contact: Moniterm, 5740 Green Circle Drive, Minnetonka, MN, 55343, (612) 935-4151.

Photon Paint Expansion Disks: MicroIllusions has released Photon Paint Expansion Disks. This package is designed to save time for the Photon Paint artist interested in using wood or marble surfaces. You don't have to start from scratch, because nearly every type of wood or marble surface is available on the two disks. Just load in the file you wish, capture a brush, wrap it on a 3D shape such as a tube or ball, or tilt it with full perspective. By using the blend mode with the dither or luminance features, you can create realistic columns of marble with proper highlights, a tile floor that has depth, or even wood paneling. The Photon Paint Expansion Disk set is available at a suggested retail price of $29.95. MicroIllusions, 17408 Chatsworth St., Granada Hills, CA, 91344. Phone: (818) 360-3715.

Prolific Inc.

Available Now!

PRO-NET



$475.00

For AMIGA™ only

PRO-BOARD

$475.00

POWER+SIMPLICITY=PRODUCTIVITY

Feel our POWER...

PRO-NET:
- *Variable template size*
- *A to E paper size*
- *Extensive Library included*
- *Auto device number with Zone control*
- *Gate swapping*
- *Single click change to negative logic*
- *Innovative weight assignment*
- *Auto page reference*
- *Dynamic error checking*
- *Creates BOM, Spare Part List, Net List, Error report, etc.*
- *Supports printers & plotters*
- *Supports Laser Printer*
- *Back annotation from PCB layout*
- *Move, Rubber...and lots more*

PRO-BOARD:
- *.025 inch grid*
- *.001 inch grid Library*
- *12 mil trace, 13 mil space*
- *Produce 1, 2, 4 layer PCB*
- *Provides silk screen*
- *Auto coordinates assignment*
- *Auto produces power and grid layers*
- *Single line auto route*
- *Optional Net List Input for guided route, no need to look at schematic*
- *Prioritized route*
- *Dynamic error checking*
- *Supports printers, plotters and Gerber photo plotters*
- *Copy, Repeat...and lots more*

See our SIMPLICITY...

Intelligent Function Keys make our programs extremely user friendly, provide maximum screen area, always display all relevant commands, avoids excessive cursor movement and screen flashing between menu & drawing, guides user through operation, minimizes training time.

ORDER OR CALL FOR DETAILS
1808 W. Southgate Ave., Fullerton, CA 92633
Tel: (714) 447-8792 Telex: 5106016526 PROLIFIC CALIF
Western Union Easy Link Mail Box 62935949

TRY OUR DEMO DISKS FOR $15 EACH.

NEW!

Also from Prolific Inc.,
4 full feature AMIGA™ Macro Cross Assemblers for Z80, 6809, 8085 & 8051.

See our SIMPLICITY

PRO-ASM

Feel our POWER
- *Includes multi-pass Assembler, Linker, and Serial Down Load*
- *Generates relocatable Object Code Module*
- *Nested Macro*
- *Includes Files*

Feel our POWER
- *Conditional Assembling*
- *Rich set of directives*
- *Global and External Variables*
- *Data format includes Binary, Motorola Hex, Intel Hex, and Tek Hex*

$85.00

DEALER INQUIRY INVITED.

AMIGA™ trade mark of Commodore Inc.

Photon Video Transport Controller: MicroIllusions has recently announced the release of Photon Video Transport Controller. Bridging the gap between computer and video tape animations, this professional software lets you regulate external video tape frame by frame controllers. It supports the Lyon-Lamb and Video Media's V-Lan systems, and may be upgraded to accommodate new controllers at any time. Transport Controller comes complete with all the functions necessary to make your work easy. For instance, it can be programmed to record from one to 300,000 frames each time an image is displayed, resulting in a smooth animation sequence.

The editor allows you to stretch short real-time animations into longer sequences. IN and OUT edit points can be set by using either the keyboard or the mouse. Transport Controller is designed to work exceptionally well with Photon Video Cel Animator, which, in turn, is highly compatible with Photon Paint. All of these fine programs are available at a suggested retail price of $299.95.

Shakespeare Upgrade: The upgraded version of Shakespeare is now shipping. Shakespeare V1.1 features improved memory management, auto column creation, alignment guides, support for foreign keymaps and alternate character sets, faster PostScript printing, support for 24-pin printers, high-quality sixteen gray-scale printing on black and white printers, and persona non grata status to a certain eastern mystic. Infinity Software will upgrade all owners of Shakespeare V1.0 free of charge. The upgrade will be sent automatically to all users who have sent in their registration card.

Shakespeare allows users to combine multiple full-colour graphics, typefaces, and text to create sophisticated colour page layouts.

With Shakespeare you can design a color brochure, produce a multi-page newsletter, create a flashy flyer, integrate charts from a spreadsheet into a business report, and turn a simple letter into a colourful document. Shakespeare provides global page layout options; full page preview mode with multipage option; on-line help; on-screen text editing, word processor-like functions; graphic cropping and scaling, 4096 colours, colour palette manipulation, and overlays in opaque or transparent mode. Shakespeare offers support for standard dot matrix, ink jet, thermal transfer, and PostScript compatible printers. Shakespeare is not copy-protected and requires 1MB of RAM. Suggested retail is $225.00. Infinity Software, 1144 65th Street, Suite C, Emeryville, CA, 940608, (415) 420-1551

External Floppy Drives for the Amiga: Memory and Storage Technology, Incorporated, announced two 3.5" 880K external floppy disk drives for the Amiga computers: the Unidrive and the Twindrive. These drives combine high capacity floppy storage with a small footprint: The Unidrive is one inch high and the Twindrive is two inches high. A switch on the rear panel allows the drive to be disabled, so the user need not unplug the external drive unit if running programs which do not execute when an external drive is connected.

The M.A.S.T. Unidrive and Twindrive require 6mA and 12mA of standby power respectively. Thus, the drive remains very cool. Both units can be powered by the Amiga or from an external power supply. In addition, the click that occurs when the diskette is removed from the drive can be disabled, and a hinged dust cover protects the heads when the diskette is removed. The Unidrive lists for $169 and the Twindrive lists for $299. Both drives are covered by a one-year limited warranty with a guaranteed factory upgrade. This means that the user can upgrade from a Unidrive to a Twindrive for only $130. Memory and Storage Technology Inc., 7631 East Greenway Road, Scottsdale, AZ, 85260, (602) 483-6359.

We had a M.A.S.T. Unidrive on here at the Transactor offices for a few months, and were very impressed with its slim design and quiet operation. Especially considering its low price, we would highly recommend this unit when an external drive is required.

Ameristar Internet Package: The Ameristar products allow the Amiga to communicate via the standard Internet protocols (TCP/IP, UDP/IP) over the Ethernet physical layer protocols. Hence the Amiga can now interoperate with most networks consisting of either TCP or Unix hosts. Many organizations can now benefit by incorporating networked Amiga systems by replacing comparably priced terminals with Amigas networked into their Unix or TCP based Ethernet backbone. The Amiga is the ideal choice for low cost, high performance graphic "front ends" to the more costly and powerful Unix workstations from Sun, Apollo and DEC to name a few.

The Ameristar Internet Package includes: AE-2000 Ethernet controller, Network File System (NFS), Telnet with VT100 emulation, Ftp, remote login (rlogin), Rsh, TCP/UDP/IP, Diagnostics and User Documentation. List price is $899 US.

NFS was originally developed by Sun Microsystems to provide transparent file sharing in a network environment. Since then, NFS has been adopted by over 100 leading computer manufacturers. Most Unix workstation vendors like Sun, Apollo and now many of the Unix implementations on various 386 platforms support NFS. Ameristar provides full client NFS support for the AmigaDOS operating system. When tied into an NFS host, your Amiga may mount and unmount Unix file systems and treat them as large and fast local disks. The remote servers appear truly transparent to you and your Amiga. Telnet provides remote network login to host computers. The Ameristar Telnet features VT100 emulation with an integrated rlogin shell. Because of the Amiga's multitasking capabilities, multiple Telnet sessions may be invoked to different hosts. This provides powerful terminal capabilities.

Rlogin provides Unix login capabilities so the Amiga can act as an ANSI Ethernet terminal. Rsh permits remote execution of Unix commands on a Host from an Amiga Client. Ftp is a remote file exchange protocol which is fully implemented and useful in non-Unix environments where NFS is not available and file transfer is required. All Ameristar Internet applications

run on top of Ameristar's optimized TCP/UDP/IP transport services. The internet layer fully supports IP fragmentation, routing, subnet masking, and provides the Application developer with an optional Unix Socket interface. Each card is shipped with diagnostics to help test and verify your Ethernet controller. A complete functional test program is included as is a send/recv test program. Additional utilities such as Chmod for permission modification, Mlist for displaying exported filesystems, and "LS" - an Ameristar version of the popular Unix command - are also included. For more information, contact: Ameristar Technologies Inc., 47 Whittier Avenue, Medford, NY, 11763, (516) 698-0834.

X11 for the Amiga: GfxBase of Milpitas, CA announces what it describes as "the price performance leader in X11 workstations". At Xhibition '88 in Cambridge, MA, GfxBase demonstrated X11 client and server implementations running native on the Amiga line of personal workstations. GfxBase showed X running on standard Amiga 2000s and Amiga 500s networked with Ethernet.

Release 1.0 of the GfxBase implementation of X for the Amiga supports monochrome displays with up to 1008 by 1024 resolution; standard X11 fonts; a variety of local clients, including uwm, xterm, xcalc, xclock and bitmap; and several communication protocols - TCP/IP, DECnet and a local message passing mechanism. The product fully supports the Amiga multitasking operating system and graphics hardware, and coexists with its native windowing environment. And, according to GfxBase president Dale Luck, "X on the Amiga provides the most cost-effective multitasking X workstation available. Additionally, since we support the Amiga's special graphics processors, the Amiga performance is comparable to a Sun 3/50 workstation." The product requires 1MB of RAM and can optionally use a three button mouse. Available first quarter of 1989 for A500, A1000, A2000. For more information, contact: GfxBase, 1881 Ellwell Drive, Milpitas, CA, 95035, (408) 262-1469. Usenet: amiga!boing!dale. BIX: duck.

NewTek releases Digi-View 3.0 upgrade: According to NewTek president Tim Jenison, "Digi-View 3.0 adds all the features that users have been asking for, including new algorithms that push the graphics chips on the Amiga harder than ever." New features include overscan with a maximum of 768 by 480 resolution. This new mode allows for borderless screen images, perfect for desktop video applications. The new half-bright mode allows for incredibly detailed images with 64 colours. The most advanced new feature, line art mode, allows you to digitize any black and white line art, logo, or other detailed two coloured image perfectly, with no extraneous picture elements. The suggested retail price for the Digi-View image capture system, including version 3.0 software is $199.95. Available from NewTek Inc., 115 West Crane St., Topeka, KS, 66603, (913) 354-1146.

FinePrint from Designlab: FinePrint is a unique printing utility that can produce almost photographic quality prints of IFF images using a normal dot matrix printer and a worn out ribbon. The program is a 'multiple shade of gray' printing program that lets you use a dot matrix printer to print images with true shades of gray. Normal Amiga graphics printouts use dot patterns to simulate shades of gray. This reduces the resolution and quality of the print. They can't reproduce highly detailed images properly. FinePrint uses a special technique to actually build up ink on every dot in the print. Using a worn out ribbon on your printer, FinePrint can get almost photographic quality prints. The Amiga printer drivers are not able to produce this type of result. FinePrint can use new ribbons also, they just don't give as many shades of gray. FinePrint lets you print images very small and very large. It can scale any size IFF image into any size print. It works with all IFF images with up to 32 colours, though it only prints in black and white. You can stop a print at any time during printing. FinePrint uses its own special printer drivers that support most popular dot matrix printers. To see if your printer is supported, call (607) 687-5740. FinePrint does not work with laser, ink-jet, or thermal printers. Available for $49.95 (plus $3.00 shipping and handling, NY residents add sales tax) from Designlab, P.O. Box 419, Owego, NY, 13827, (607) 687-5740.

Adrum from Haitex Resources: Adrum is a full-featured four-voice drum machine capable of stereo output using sound samples loaded from disk in IFF format. Adrum can have 26 sounds in memory at once (dependent on system memory) to produce a totally definable 'kit' from which to create rhythms. Once a kit is created, the sequencer may be programmed. Up to 64 different measures of variable length (0to 64 beats/measure) may be defined. Adrum has editing features not found on 'standard' drum machines. Selective removal, replacement, measure copy, and delete/insert commands are all available pull-down menus. Once created, a song can be replayed, and if a MIDI interface is attached, Adrum can synchronize to or be controlled by other musical instruments on the MIDI system. Adrum is published by Bullfrog Productions, and is being marketed in North American by Haitex Resources Inc., 208 Carrollton Park, Suite 1207, Carrollton, Texas, 75006, (214) 241-8030. Suggested retail is $79.98 (US).

EZ-BACKUP from EZSOFT: EZ-BACKUP is a new kind of hard disk backup program which manages the space on the backup floppy disks. EZ-BACKUP allows you to save from 0 to 255 versions of each file. By deleting obsolete files, EZ-BACKUP allows you to use the same set of floppy disks over and over again. Only one full back-up is ever required. EZ-BACKUP completely eliminates the necessity of having to do full backups each time the incremental backup disks begin to pile up. Other program features include: optional warning screen reminds users to backup their data; volume label checking prevents writing over important files; archive files in standard Amiga format, works with all standard utilities; individual file recovery provides mouse-oriented program for recovery of individual files; multitasking; provides archive-bit utilities. EZ-BACKUP is not copy-protected. EZSOFT also offers free upgrades and telephone support. Available for $49.95 from: EZSOFT, 21125 Chatsworth Street, Chatsworth, CA, 91311, (818) 341-8681.

Hack Lite from The Software Distillery: What's the difference between Hack Lite, new from the Software Distillery, and the previous versions of Hack? Why should you bother to switch? Well, Hack Lite it faster, more flexible, fairer, and funnier.

Major parts of Hack Lite have been completely rewritten to improve its speed. You should no linger expect the annoying slowdowns of the past, even when you're running from a swarm of bees or a squadron of Orcs.

Even more than before, Hack Lite has been modified to fit the Amiga, not just to be another program copied from Unix with no thought to its new home. The menu interfaces have been improved and generalized. (But, of course, the old command interface is still there for those who prefer it.) Additionally, support has been added for keeping multiple save files, for saving on disks other than the game disk, and for making effective use of the RAM disk.

Hack Lite is fairer than previous versions of Hack. By this, we mean that we have tried to keep the game consistently challenging, without either making it impossible to progress beyond a certain point, or allowing it to become boring.

Most of all, Hack Lite is funnier. New features, like Keystone Kops, the pray command and those pesky rock moles, have been designed in the absurd tradition of fortune cookies and the Scroll of Punishment. Hack is a game of brutal violence and lingering death, but that's no reason not to meet your fate giggling. Even when you're trapped in the morgue at midnight, surrounded by vampires, even though your dog has gone mad and a nymph has stolen your last clove of garlic, cheer up! You've still got one pie left to throw.

Hack Lite requires 512K and 1 disk drive. It is available from your local Freely Redistributable Disk dealer. Or order it from: The Amigans, Box 411, Hatteras, NC 27943 ($6.00 a disk).

Benchmark Modula-2 from Avant-Garde Software: Benchmark Modula-2 is an easy to use interactive programming environment featuring an integrated editor, compiler, linker and loader. The user can: perform complex editing tasks at turbo speed with the EMACS based editor which has over 150 commands (The editor supports multiple files and windows.); compile the current file at a rate of 10,000 lines per minute with bursts of up to 30,000 lines per minute; find next error in the source file; link most programs in less than five seconds. Three support modules are also available from Avant-Garde.

Simplified Amiga Library: Modules for creating and using screens, windows, gadgets, menus, IDCMP, console device, double-buffered screens, graphics elements, and speech synthesis.

IFF & Image Resource Library. This package includes a library of functions for reading and writing Interchange File Format (IFF) files. To support the addition of new data types, low-level IFF procedures are implemented fully. Procedures are defined for compression and decompression of data. High-level procedures for reading or writing a complete IFF file are defined. The complete source to the IFF library is included. Any number of IFF pictures or brushes may be attached to an executable file as a resource; the images will be automatically loaded into CHIP memory when the program is executed. Each image attached to the executable file may be converted to one of three popular formats: Image or Bob format, Simple Sprite format and Virtual Sprite format. The imagery for a program may be modified without recompiling or relinking. Separate data files for storing the graphics imagery become unnecessary.

C Language Library. Programmers familiar with C can use many of the familiar functions in their Modula-2 programs. This library includes a large set of functions for performing buffered and unbuffered file I/O (fopen, fclose, getc, fseek, open, close, read, write), formatted output (printf, fprintf, sprintf), input conversion (scanf, fscanf, sscanf), dynamic memory management (malloc, calloc, free), memory operations (setmem, repmem, swapmem), string functions (strcat, strncpy, strlen).

Retail price of each support module is $99.95. Benchmark Modula-2 is $199.95. The official source for Benchmark Modula-2 is now: Avant-Garde Software, 2213 Woodburn, Plano, TX, 75075, (214) 964-0260.

Interchange from Syndesis: Share objects between Sculpt 3D, VideoScape 3D, Forms In Flight, and Turbo Silver. Convert from any format, to any format, in seconds. Use any 3D program to create objects, then InterChange them to another 3D program for ray-tracing or animation. The easy-to-use InterChange master program converts colour and texture information, too.

The InterChange system is expandable into the future. All InterChange Conversion Modules are separate, multitasking programs. They all work together, and you can add new modules to support new 3D formats. You get the InterChange master program and both the Sculpt 3D and VideoScape 3D Conversion Modules for just $49.95 (US). The Forms In Flight and Turbo Silver Conversion Modules are sold separately for $19.95 each. Famous Amiga artists made lots of 3D objects for Sculpt 3D and VideoScape 3D. Object Disk #1 includes a full font of block letters, an Amiga logo, a space shuttle, office equipment and much more, for just $19.95.

InterFont from Syndesis: 3D object font designer. Turn your 3D modeling program into a video titling system. InterFont is a new type of font editor. It lets you create fonts that can be used in 3D animation and modeling programs, in conjunction with InterChange and the InterFont Conversion Module. With InterFont, you can quickly create 3D logos and 3D video titles for animations. You create fonts which can be converted to formats used in 3D modeling programs such as Sculpt 3D, VideoScape 3D, Forms In Flight and Turbo Silver. InterFont can also create structured drawings in Aegis Draw format which can then be imported into Professional Page. InterFont comes with pre-made fonts too - so you can get off to a quick start. Create whole blocks of text for ray-tracing, structured drawings or animations. For more information about InterChange or InterFont, contact: Syndesis, 20 West Street, Wilmington, MA, 01887, (508) 657-5585.

Reader Service Card

Fill out this card (please print) or attach your peel-off mailing label, and circle the reader service number for the ads that you'd like more information about (see Advertiser Index below).

Name _____

Company _____

Address _____

_____

Postal/Zip Code _____

1	2	3	4	5	6
7	8	9	10	11	12
13	14	15	16	17	18
19	20	21	22	23	24
25	26	27	28	29	30

TA V1I5

Advertiser	Page	Reader Service #
Amiware	59	1
ASDG Inc.	36	2
Briwall	18,19	3
Central Coast Software	1	4
Comp-U-Save	37	5
Comspec	Inside Back Cover	6
Creative Computers	9-11	7
Discovery Software	Outside Back Cover	8
Electronics 2001	36	9
Emerald Intelligence	Inside Front Cover	10
Intercomputing Inc.	27	11
Interface Technologies	71	12
MCP Associates Inc.	36	13
Micro Systems International	17	14
Palomax Inc.	65	15
Prakticon Inc.	65	16
Prolific Inc.	77	17
Redmond Cable	65	18
The Memory Location	17	19
UEdit	65	20
William S. Hawes	17	21

Advertising in Transactor for the Amiga

Interested in advertising in *TransAmi?* Call Max Jacobson at 416-764-5274 NOW!

place
stamp
here

Transactor
10-85 West Wilmot Street
Richmond Hill, Ontario, Canada
L4B 1K7

Introducing our new...

TOLL-FREE ORDER LINE
1-800-248-2719 Extension 911

Here's a convenient way to order magazine or disk subscriptions and other products. Nothing to fill out or drop in the mail! Just call the toll-free number above and have your credit card number ready. Available in the United States only.

NOTE: The toll-free number can be used for ordering only; for subscription queries, etc., call our office directly.

for the AMIGA Transactor

The Magazine for Amiga Programmers

Please print your name and address below (or transfer your mailing label)

Check here for change of address ☐

Name: _____

Address: _____

City: _____ *Province/State:* _____

Zip/Postal: _____ *Phone:* _____ *CompuServe ID:* _____

NOTE: Prepayment required. Purchase orders will be accepted ONLY if accompanied by payment.

Subscribers only: please enter subscription number from mailing label ☐

Subscription Order Section

PLEASE NOTE THAT TRANSACTOR FOR THE AMIGA IS NOW PUBLISHED MONTHLY

	Canada ($Cdn.)			U.S.A. ($US)			Foreign ($US)			Air Mail ($US, overseas only)		
	Mag	Disk*	Both*	Mag	Disk	Both	Mag	Disk	Both	Mag	Disk	Both
12 issues	☐ 35.00	☐ 123.00	☐ 158.00	☐ 27.50	☐ 101.00	☐ 128.50	☐ 38.50	☐ 119.50	☐ 157.50	☐ 73.00	☐ 145.00	☐ 218.00
24 issues	☐ 65.00	☐ 220.00	☐ 270.00	☐ 49.00	☐ 160.00	☐ 190.00	☐ 69.00	☐ 220.00	☐ 290.00	☐ 135.00	☐ 250.00	☐ 390.00

*** Ontario residents please add 8% of disk subscription price for provincial sales tax. No PST on magazine subscriptions.**

☐ **Please send Microfiche instead of magazines** ☐ **Send me info on Transactor Classic (C64/128)**

Combo Subscription Bonus!

If you are subscribing to BOTH the magazine and the disk, check one of the following FREE gifts:

☐ **Transactor Classic Amiga Disk #1** ☐ **Fish Catalog Disk** ☐ **Extended Subscription (7 Issues)**

Product Order Section

Ontario residents please add 8% PST on all items

	$Cdn.	$U.S.	
☐	14.95	12.95	Transactor Classic Amiga Disk #1
☐	14.95	12.95	Transactor Classic Amiga Disk #2

Fish Disks $6.95 US, $7.95 Cdn. each

Note: Disk numbers not yet released may be ordered in advance - simply add numbers to end of list. Disks 57, 80 and 88 are not available.

☐ 1	☐ 2	☐ 3	☐ 4	☐ 5	☐ 6	☐ 7	☐ 8	☐ 9	☐ 10	☐ 11	☐ 12	☐ 13	☐ 14
☐ 15	☐ 16	☐ 17	☐ 18	☐ 19	☐ 20	☐ 21	☐ 22	☐ 23	☐ 24	☐ 25	☐ 26	☐ 27	☐ 28
☐ 29	☐ 30	☐ 31	☐ 32	☐ 33	☐ 34	☐ 35	☐ 36	☐ 37	☐ 38	☐ 39	☐ 40	☐ 41	☐ 42
☐ 43	☐ 44	☐ 45	☐ 46	☐ 47	☐ 48	☐ 49	☐ 50	☐ 51	☐ 52	☐ 53	☐ 54	☐ 55	☐ 56
☐ 58	☐ 59	☐ 60	☐ 61	☐ 62	☐ 63	☐ 64	☐ 65	☐ 66	☐ 67	☐ 68	☐ 69	☐ 70	☐ 71
☐ 72	☐ 73	☐ 74	☐ 75	☐ 76	☐ 77	☐ 78	☐ 79	☐ 81	☐ 82	☐ 83	☐ 84	☐ 85	☐ 86
☐ 87	☐ 89	☐ 90	☐ 91	☐ 92	☐ 93	☐ 94	☐ 95	☐ 96	☐ 97	☐ 98	☐ 99	☐ 100	☐ 101
☐ 102	☐ 103	☐ 104	☐ 105	☐ 106	☐ 107	☐ 108	☐ 109	☐ 110	☐ 111	☐ 112	☐ 113	☐ 114	☐ 115
☐ 116	☐ 117	☐ 118	☐ 119	☐ 120	☐ 121	☐ 122	☐ 123	☐ 124	☐ 125	☐ 126	☐ 127	☐ 128	☐ 129
☐ 130	☐ 131	☐ 133	☐ 134	☐ 135	☐ 136	☐ 137	☐ 138	☐ 139	☐ 140	☐ 141	☐ 142	☐ 143	☐ 144
☐ 145	☐ 146	☐ 147	☐ 148	☐ 149	☐ 150	☐ 151	☐ 152	☐ 153	☐ 154	☐ 155	☐ 156	☐ 157	☐ 158
☐ 159	☐ 160	☐ 161	☐ 162	☐ 163	☐ 164	☐ 165	☐ 166	☐ 167	☐ 168	☐ 169	☐ 170	☐ 171	☐ 172

The Transactor for the *AMIGA* Disk $12.95 US, $14.95 Cdn. each

(these disks correspond to issues of **Transactor for the *AMIGA***)

☐ Vol. 1, Iss. 1 ☐ Vol. 1, Iss. 2 ☐ Vol. 1, Iss. 3 ☐ Vol. 1, Iss. 4 ☐ Vol. 1, Iss. 5

Add flat $2.00 shipping & handling for any number of items ordered from above box.

Transactor for the *AMIGA* Back Issues and Microfiche: $4.50 U.S., $4.50 Cdn. (MF = Microfiche)

☐ V1, #1 ☐ MF ☐ V1, #2 ☐ MF ☐ V1, #3 ☐ MF ☐ V1, #4 ☐ MF ☐ V1, #5 ☐ MF

Ontario residents please add 8% provincial sales tax for any items ordered from this box.

☐ Visa ☐ MasterCard ☐ Cheque/MO enclosed Cheque #_____ Dated ____/____/____ Amount _____

Acct. # _____ Expires ____/____ Signature _____

Please allow 6-8 weeks for delivery in the U.S., 4-6 weeks in Canada.

FOLD ALONG SCORES WITH CORRECT REPLY CARD ON OUTSIDE. USE TAPE, PLEASE - NO STAPLES OR GLUE.

NO POSTAGE NECESSARY IF MAILED IN THE UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 390 BUFFALO, NY

POSTAGE WILL BE PAID BY ADDRESSEE

The Transactor
P.O. Box 338 Station 'C'
Buffalo, NY, 14209-9990

Business Reply Mail
No Postage Stamp Necessary
If mailed in Canada

Postage will be paid by:

Canada Post Postes Canada
46487

The Transactor
501 Alden Road
P.O. Box 3250
Markham Industrial Park
Markham, Ontario
L3R 9Z9

COMSPEC'S SEVEN BITS TO SPICE UP YOUR BYTES

MCS 1000
Connect any IEEE type printer to virtually any computer utilizing standard Centronics output. Plugs into a parallel port on your computer.

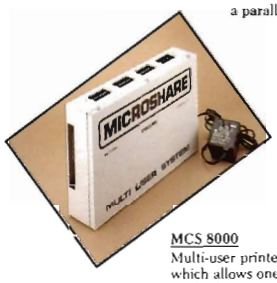
AX 2000
1 or 2 MB FAST RAM for the Amiga 1000 computer. Provides the extra RAM necessary to run advanced applications quickly and efficiently. Runs with no wait states. Complete with non-volatile RAM disk.

SD-20 SD-40 SD-60
Auto-booting external hard drives for the Amiga. The Comspec Hard Drive system consists of a SCSI Host Adaptor * and a Hard Drive Chassis, available in 20, 40, and 60 MB sizes. The only Hard Drive system for the Amiga that provides auto-booting under WB 1.2.

* Also available separately.

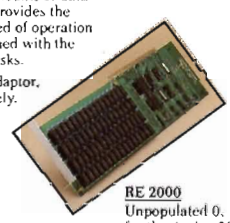
BN 20
Bernoulli 20MB 5 1/4" removable cartridges allow you to store sensitive data in a secure place. Also allows for the sharing of large amounts of data between computers. Provides the storage space and speed of operation of a hard drive combined with the portability of floppy disks.

Requires SCSI host adaptor, also available separately.



MCS 8000
Multi-user printer network/buffer which allows one printer to do the work of eight. Fully compatible, the MCS 8000 allows up to eight computers to share one printer. A must for any multi computer environment.

MCS 6550
256K print buffer. Frees up computer time by allowing you to process and print at the same time. Works alone or can be combined with the MCS 8000 to increase memory. No installation is required.



RE 2000
Unpopulated 0, 1 or 2 MB FAST RAM for the Amiga 2000. Similar to the AX2000 with 1/6 power consumption of competitive products. This highly reliable memory board is compatible with all standard Amiga hardware and software.

Available from:

COMSPEC

74 Wingold Ave., Toronto, Ont., M6B 1P5 (416) 785-3553 FAX 785-3668

Distributed worldwide by:

Precision Software Ltd.
6 Park Terrace
Worchester Park
Surrey KT4 7JZ
England
01 330-7166
Fax 01 330-2089

Microton
Bahnofstrasse 2
Postfach, 40 Ch 2542
Pieterlen, Switzerland
032 872429
Fax 032 872482

Data Stream
29 Carlyle St
Napier, New Zealand

Run Informatique
62 Rue Gerrard
Paris, France
45 01 51 44

Airstat of Canada Inc.
3575 – 14th Ave.
Markham, Ont. Canada
L3R 2L9
1-800-269-8314
Fax (416) 470-6805

Interest A.S.
Markveien 44B
P.O. Box 59
4030 Hinna, Norway
(04) 851036
Fax (04) 851959

*Specifications subject to change without notice.

SWORD OF SODAN™

Grasp the Sword of Sodan and feel the true power of arcade action!

Creaking, groaning, shrieking, teeth clenching arcade realism. Deep dark forests. Misty, eerie graveyards. Treacherous, guard-filled streets. Dare to enter the evil castle, Craggamoore. Wield your sword through its winding corridors and rank rotting dungeons. Your story will be the greatest ever told — Your reward will be the greatest ever known!

FEATURES:

- ***Battle with massive animated characters that stand 2/3rds the size of the screen.***
- ***Fight as Hero or Heroine through 11 incredibly challenging levels!***
- ***Study your past performance in Replay Mode! (extra memory required)***

Packed into 3 disks with more than 4 MEGABYTES of detailed graphics and digitized sounds!

Available for Amiga™, Coming soon for IBM® and compatibles, Apple II GS.

AMIGA screens shown.

DISCOVERY SOFTWARE

Fight your way to your local software retailers. Or, call 1-800-342-6442 for VISA and MasterCard Orders. For check or money orders, send $49.95 plus $3 shipping and handling to DISCOVERY SOFTWARE INTERNATIONAL INC. 163 Conduit Street, Annapolis, MD 21401. Allow 4-6 weeks for delivery. Sword of Sodan is a trademark of Discovery Software International, Inc. Amiga is a trademark of Commodore Amiga, Inc. IBM is a registered trademark of International Business Machines, Inc.

Dorian 1988 ©